Email-based Intelligent Virtual Assistant for Scheduling

Written by

Anand Chowdhary

Supervised by

Dr. Job Zwiers

Creative Technology BSc

University of Twente, Enschede, the Netherlands

Abstract

Manually setting up appointments by email wastes tens of hours every month for professionals, because several email exchanges are required before receiving confirmation. Since not everyone can afford to hire full-time assistants, Email-based Intelligent Virtual Assistants (EIVA) can help by automating this task.

In the research described in this paper, a functional EIVA was developed based on research and industry best-practices. Users could simply add their assistant's address as 'CC' in an email, and EIVA would share the recommended location and time slots with guests automatically, based on the user's availability (determined using their calendar) and scheduling preferences. A companion web application was also developed to manage meetings and settings. The code is open source and was written in TypeScript with Node.js for the backend and Vue.js for the frontend, and deployed on Amazon Web Services architecture in Europe. The product was built with a focus on data privacy and user personalization, through a series of feedback cycles with the external client.

A user experience evaluation of EIVA was conducted with 30 participants that found positive reception. The average rating of the overall assistant was calculated to be 4.4 out of 5, and that of the web app was 4.5 out of 5. Users' behavior was also understood with the help of heatmaps and visualizations using pageview and mouse clicks tracking. All but one participants said that EIVA met their expectations, and 25 out of 30 would use it in the future if it launches as a service. Most would also be willing to pay for it, with an average amount up to €6.16 per month. Participants also shared their frustrations and feature recommendations. In the future, natural language processing-based classification should be improved and user recommendations can be implemented before launching EIVA as a service for consumers.

Acknowledgements

First and foremost, I would like to thank Dr. Alma Schaafstal, Program Director of Creative Technology, for her continued support, guidance, and mentorship during the past three years.

I would like to thank Dr. Job Zwiers, my thesis supervisor, without whom this project wouldn't be possible, and Dr. Champika Manel Epa Ranasinghe, the critical observer for this project, for her valuable feedback.

I would also like to thank the client, Speakup B.V., for their generous support of this project. I am particularly grateful to Florian Overkamp, who has been with me every step of the way — from helping foster the original idea in 2017, to funding multiple projects over the years, and finally helping shape this graduation project and service.

Thanks to my team at Oswald Labs, especially my cofounder Mahendra Singh Raghuwanshi, without whom I wouldn't be able to sustain building a startup while graduating CreaTe. As part of our accelerator program, Oswald Labs also sponsored the cloud credits required for this project.

I am extremely grateful to my family and friends, especially my parents, my brother, and my sisters, for their continued support. I also want to thank my girlfriend, Sukriti Kapoor, for sharing the survey with her colleagues, and for proofreading this thesis.

Finally, I would like to thank the Creative Technology faculty and staff — Dr. Dennis Reidsma, Dr. Katarzyna Zalewska, Dr. Erik Faber, Dr. Khiet Truong, Dr. Jefferey White, Dr. Edwin Dertien, Chris Vermaas, Richard Bults, and Alfred de Vries — and the Twente startup ecosystem — Michael Angelo Groeneveld, Mike Verkouter, Peter Langela, Emiel Pegge, Thomas Mensink, Gilles Meijer, Rick Sulman, and Amy de Lange — for helping shape my entrepreneurial spirit.

Contents

List of Tables

List of Figures

1 Introduction

Setting up appointments by email is a non-trivial waste of time. Last year, 110 billion consumer emails were sent per day, with a very common use case being setting up appointments [4]. Several email exchanges are required in order to find a suitable time and place where all parties are available, and almost 1 in 5 users struggle with finding sufficient available time slots [5]. This back-and-forth calendar conflict resolution on average waste 17 minutes per meeting [6]. Moreover, over 3 in 5 meetings end up getting rescheduled for a different time or place, which wastes additional time in confirmations [7]. This adds up to about 9 hours wasted every month per person.

For some professionals, scheduling these meetings manually can feel like a "frustrating distraction from the things that matter," so much so that they hire assistants to help with the task [8]. However, not everyone can afford full-time assistants and will therefore turn to software solutions.

An intelligent virtual assistant that can schedule appointments over email can be a faster and more affordable alternative to manually scheduling appointments. In this paper, such an assistant is proposed — Email-based Intelligent Virtual Assistant (EIVA /iːvɑ/). EIVA[1] can access a user's calendar and find empty meeting slots based on their location and scheduling preferences. It can then send and respond to emails regarding scheduling meetings on the user's behalf, and the user can directly interact with EIVA over email.

The goal of the research described in this paper is to:

1. Build a functional virtual assistant with email integration, along with a companion web application to manage meetings and preferences; and

2. Conduct a user evaluation on both the usability of the assistant over email and the user interface of the web application.

Finally, as the external client is a communications technology company, the product is built with best practices in mind, including using a modern stack of front-end and back-end programming languages and frameworks, with a focus on customizability, privacy, and security.

[1]In this paper, the term 'EIVA' is not only used as an acronym, but also as a proper noun for the personification of the virtual assistant.

2 State of the Art

2.1 Scheduling

2.1.1 Calendars

In their fundamental forms, calendars are the “most basic of all human sense-making devices” [9]. From time immemorial, calendars have been used as tools to establish patterns through which nearly all institutions, societies, and social groups manage orderliness. Notably, the first “book” printed by Gutenberg in 1452 was a calendar [10], and today, professionals increasingly use digital calendars for the same purpose of time management and accountability. Digital calendars have become “the standard office tool for coordinating and synchronizing work activities” [9].

There are several calendaring software products available to professionals, both commercial and open source. In general, they offer a range of features, including calendar views, built-in address book, appointment reminders, and email integration [11]. Many major internet companies offer such solutions, such as Google Calendar, Microsoft Outlook, Apple Calendar, and Mozilla Thunderbird. For company-wide adoption, enterprise software such as Microsoft Exchange and IBM Notes are also available [12]. The University of Twente, for example, recommends the use of Google Calendar [13].

However, company-wide adoption of scheduling software is troublesome. For end users, it requires behavioral change, such as using a different tool than they are used to [14]. Additionally, there is no “one size fits all” solution to calendaring, because there is a wide variation in people’s meeting needs. For example, a professor who wants to schedule office hours with students might want students to pick their preferred meeting times, whereas a corporate executive might want to heavily control their availability times and response to meeting requests.

Professionals who can afford to hire assistants choose to delegate the hassle of scheduling. In a survey of administrative assistants, all but one reported that most of their work was scheduling-related, wasting hundreds of man-hours [15].

2.1.2 Shortfalls of Calendaring Tools

Currently available products force the end user to manually enact the process of scheduling [16]. For this reason, these tools avoid the need to actively reason about constraints and the user’s scheduling preferences, such as which one of the many available meeting times to choose when underconstrained, and how the user would prefer to relax a meeting request when overconstrained [17]. In a work environment, there are additional decisions to make, such as which conference room is best for a given type of meeting, or setting up remote meetings using video calling software. All of these decisions are presently passed onto the end user.

These calendaring tools are also “particularly cumbersome when attendees need to coordinate multiple schedules while each using their own individual tool and approach” [8]. This is especially true for users who have multiple calendars, such as a different calendar for work and personal lives. A survey of 23 professionals found that substantially more than half the respondents maintain more than one calendar, with some respondents using as many as six calendars at once [18]. Another survey reported that professionals used 2 calendars on average [19]. Keeping separate work and private calendars is the norm "even though the standard weekly interface of most scheduling software is meant to occlude such distinctions" [9]. This suggests that an important feature of the ideal scheduling software would be the ability to synchronize and find availability from multiple calendars.

Apart from calendaring tools, some users opt for scheduling tools that have additional features available, though they too possess shortcomings. For example, Google Calendar's "Find a time" feature helps users select a time by displaying the availability of all participants graphically [20]. This is particularly helpful when integrated on an institutional level, with features such as room booking and team invitations, but it also introduces new challenges for participants who do not add their agenda to their calendar well in advance. A 1982 survey found enormous differences in the time span coverage between respondents, with some users planning appointments months in advance while others concerned only with the current day and the day following [18]. Hence, scheduling software must be conversational and confirm the proposed time with guests without directly relying upon their calendar.

Although digital calendars have been the standard office tool for professionals, they are currently unable to bridge the communication gap between the end user and the invitees. Current scheduling products also lack a holistic solution to calendaring due to a wide variation in people's meeting needs. Thus, end users have to rely upon emailing to communicate meeting details effectively, wasting otherwise productive man-hours.

2.1.3 Smart Scheduling Tools

Section 2.1.1 highlights that although there is a multitude of scheduling softwares available, they are not email-based intelligent virtual assistants and therefore, lack the potential benefit that personalized AI technology could bring. Today, there are early products available in the market that have some features of EIVA, like automatically responding to emails and recommending meeting times based on multiple calendars.

Scheduling Interfaces The first "smart" scheduling tools took the form of applications where users could enter a few possible times into an online calendar and guests would select the slots they preferred. Companies such as MeetOMatic and MeetMax were early pioneers in this field, founded in 1998 and 2003 respectively. In the Netherlands, Datumprikker is the most popular group scheduler, with almost 20 million unique users.

For one-on-one meetings, products such as Calendly are used by consumers. Similar to Datumprikker, users can enter their preferred time slots or connect with their calendars, and guests can select their preferred times. Other products in this space are Doodle, TimeTrade, and Mixmax.

However, the same problem of "app fatigue" applies to these types of schedulers. According to Richardson, there is "no time in the lives of busy professionals for yet another finicky computer program; what people really needed was a machine that worked just like a human assistant" [21].

Agent-based Schedulers Microsoft's Cortana, though primarily a speech-based assistant targeted towards consumers, has calendaring and scheduling features as well. Cortana has built-in Office 365 capabilities, Microsoft's enterprise solution that includes integrated email accounts with calendars. Apart from being available on mobile devices, Cortana is also built into Windows 10, which has over 400 million users. Similarly, Amazon's Alexa and Apple's Siri also have scheduling features. However, these can merely send calendar invitations but cannot automate the email back-and-forth process.

There are currently two major products in the market with support for back-and-forth emails. The leader in the AI-powered scheduling assistant industry is x.ai, a New York-based startup founded in 2014 that has raised over $40 million in venture capital [22]. Using the service, customers can select between Amy and Andrew, a female and male virtual assistant respectively, which send send emails with proposed times and confirm appointments.

The main competitor to x.ai is Clara by Clara Labs, a similar virtual assistant for scheduling over email [23]. Clara can also send email reminders to confirm attendance and find available conference rooms in an office. However, Clara employs a hybrid approach — when it has a high confidence in its proposed response, it responds automatically, but falls back on humans in other cases.

There are also highly specialized solutions available. For example, in the travel industry, Mezi uses a chatbot interface to ask the user questions, and can book their flights and hotels automatically and add them to the user's calendar. In the high-tech space, Duplex from Google uses advanced neural text-to-speech and speech recognition to place real phone calls to businesses to schedule appointments.

2.2 Web Development

For EIVA, a web application and an email interface have to be developed. JavaScript enables interactive web pages and is one of the core technologies of the web [24]. The vast majority of websites use it for client-side page behavior. Since JavaScript can also be used for server-side execution (using a runtime such as Node.js or Deno), it is the ideal choice for full-stack web development.

However, building large-scale applications with JavaScript is challenging because of loose types that result in otherwise preventable errors. This has led to demand for custom tooling. Today, there are many competing languages that compile to JavaScript, including TypeScript, ClosureScript, CoffeeScript, Elm, PureScript, and others.

According to the annual survey The State of JavaScript, out of all the languages that compile to JavaScript, TypeScript is the language of choice for most developers. The number of developers who responded that they have previously used TypeScript and would like to use it again has grown from 20.8% in 2016 to 58.5% in 2019. Only 12% of all respondents in 2019 had not heard of TypeScript or were not interested in learning it [25].

TypeScript is developed and maintained by Microsoft and is a strict syntactical superset of JavaScript that adds static typing to the language [26]. Designed for large applications, TypeScript transcompiles to JavaScript and can be used for both client-side and server-side execution, which makes it a great fit for the development of EIVA [27].

2.2.1 Backend

To execute JavaScript on the server, a runtime environment is required. Node.js is the most popular such runtime and has an event-driven architecture capable of asynchronous I/O [28]. This makes Node.js optimized for scalability and especially useful for real-time applications.

TypeScript features such as interfaces and decorators are extensively used in the codebase. For example, the `@Controller` decorator is used to describe the route of a controller, which is then injected in the Express application.

```
import { Controller, Post } from "@staart/server";
@Controller("example")
export class ExampleController {
  @Post()
  postExample() {
    return { success: true };
  }
}
```

The above example generates an API endpoint of `/v1/example` (with versioning enabled, Section 6.1.6) which accepts `POST` requests and responds with a JSON object. This generator makes development and understanding the source code much clearer.

2.2.2 Frontend

When it comes to choosing a frontend framework for JavaScript web apps, the main options are React, Angular, Ember, Vue.js, and most recently, Svelte [29]. Although Svelte is a relative newcomer (released in 2016, compared to Vue.js in 2014 and React in 2013), it has been adopted rapidly by the community. However, it is still not production-ready and therefore risky to use when developing the EIVA frontend app [30]. Furthermore, in the State of JavaScript 2019, only 75% of all developers were aware of Svelte, whereas all 100% were aware of React, Angular, and Vue.js [25].

In terms of developer interest, Vue.js and React were rated 64% and 61% respectively, whereas Angular and Ember were much lower at 23% and 18% respectively. However, React is developed by Facebook, while Vue.js is built by a community of open source developers, and both have near-universal support in the frontend ecosystem. Therefore, Vue.js was selected as the frontend framework for EIVA; it offers an easy way to use HTML-based template syntax and single-file components [31].

2.2.3 Email Interfacing

Sending emails may seem like a trivial task, but requires many steps to ensure delivery. For example, outbound emails must be authenticated with a Domain-based Message Authentication, Reporting and Conformance (DMARC) policy to indicate that they are protected by Sender Policy Framework (SPF) and DomainKeys Identified Mail (DKIM). If not signed, emails are often rejected or marked as 'Spam' in most clients. Typically, a transactional email service such as the Simple Email Service (SES) from Amazon Web Services (AWS) is used, which provides a simple API and handles the signing and sending of emails.

On the other side, for receiving emails on a domain, Email Exchange (MX) records are added to the DNS server. In this case, AWS SES can also receive emails and store them in a specific location. Both of these processes are essential for email-based scheduling, and their implementation is described in Section 6.1.9.

2.3 Author's Previous Work

EIVA is the evolution of a project the author and the client envisioned a few years ago. In October 2017, the client (Speakup B.V.) organized a hackathon in Enschede, and the author participated with a colleague to build a concept of a disruptive communication technology. During the 24-hour period, Ara, a chatbot app was developed that could send scheduling messages to team members also using the app. Two prizes worth €500 each were awarded for this project. Although Ara was only a prototype of an app-based assistant, it started a relationship with Florian Overkamp, the founder and chief entrepreneur of Speakup, and it was decided to take the idea forward in the coming months.

During the summer of 2018, the researcher worked closely with Speakup on materializing Ara. For a period of two months, the idea of scheduling assistants was expanded, and it was decided to focus on email as the primary form factor. At the end of the project, a functional prototype of an assistant was developed that could send appointment confirmation emails. This used "Wizard of Oz AI" to understand natural language, primarily built using 'if' statements.

For a few months, Ara was used by the researcher to test the assistant's email confirmation feature. In fact, in a 2017 interview titled 'UT student among the top 50 young entrepreneurs' with UToday, University of Twente's independent publication, Jelle Posthuma stated the following:

> It was easy to schedule an interview with the first-year student of Creative Technology. His “personal assistant” Ara sent a friendly email replying: “You’ll be welcome to come next Wednesday.” Chowdhary is a co-owner of the company Oswald Labs, which develops products for people with disabilities. His office is in Roombeek. “Anand, you’re working hard: you even have a personal assistant. . . ” A big grin appears on the young student’s face. “Yes, I built her myself. Ara is a computer. Her AI recognizes emails and schedules my appointments.” [32]

Although Ara didn’t have any AI at this point, it was a successful proof-of-concept. During the summer of 2019, the researcher and the client worked together on building on top of an open source framework built for Software-as-a-Service (SaaS) companies built by the researcher (Staart API and Staart UI) [33]. This is the underlying framework that EIVA is built on top of.

Finally, during this Graduation Project, the researcher and the client continued working together, and EIVA is a direct successor to Ara. However, this thesis presented the opportunity to take a research-driven approach, and no earlier code was reused, resulting in a new product built from the ground up with more sophisticated languages and frameworks.

Table 1: Comparison of Ara and EIVA

Feature	**Ara**	**EIVA**
AWS infrastructure	✓	✓
Node Natural for NLP	×	✓
Language	JavaScript	TypeScript
Node.js Runtime	v9.x	v12.x–v14.x
Full-stack framework	×	Staart
Support for teams	×	✓
Developer API	×	✓
Data export/delete	×	✓
Incoming email logs	×	✓
Clearbit integration	×	✓
iCalendar URL support	×	✓
Two-factor authentication	×	✓
open source license	MIT	SSPL

3 Methods and Techniques

After building the assistant service and companion web app, a user experience evaluation is conducted. In this standard Human Media Interaction (HMI) research, both quantitative data and qualitative data is recorded.

3.1 Data Collection

Structure After building and deploying a functional virtual agent and its companion web application, a user experience research is conducted with participants with diverse backgrounds. The survey is divided in three parts:

1. Introductory Survey
2. Product Usage
3. User Experience Evaluation

Sampling For participant selection, voluntary response sampling was used with participants in the close network of the researcher and client. A link to an online survey was sent in the internal communication systems of Speakup B.V. and Oswald Labs, and was also posted on social media. Since the link was posted to social media, it is hard to calculate a response rate since potentially thousands of users could have seen it.

Question Design In the Introductory Survey, questions were designed as multiple-choice questions that are quick to answer. The questions were divided into three categories: (i) general information about the participants such as their profession and their email usage; (ii) assistants; and (iii) scheduling preferences. In the user experience evaluation, a combination of long text answers, rating scales, and multiple-choice questions was presented.

Questions are specifically written in a first-person form, similar to the way the assistant interacts, and additional information has been included when asking for potentially personal details. Options in multiple choice questions are randomized to eliminate order bias.

Surveying Method A link to an online form was shared with participants; there were no in-person meetings, and there was no way to interact with the researcher while participating in the research. However, users could share their feedback with the researcher using both the form and the website. Google Forms was used to create the form consisting of all three parts, prefixed with the Information Brochure and Consent Form.

Participant Observation Participants were "observed" when they used the web app using automated tracking systems. For example, all pageviews, API requests, and clicks were anonymously logged with users' consent, which helped understand how users used the app. The results are visualized in Section 7.2.3.

3.2 User Experience Research

3.2.1 Introductory Survey

After reading the Information Brochure (Appendix 1) and signing the Consent Form (Appendix 2), participants answer 16 questions about their prior experience with scheduling appointments, email usage, and personal assistants. This information helps with understanding how users currently set appointments, and with determining the default values for user preferences (Section 5.2.2). The Introductory Survey is divided into two parts; the first part is focused on scheduling preferences and email usage:

1. Which type of profession best describes you? This helps us relate scheduling experience with types of work.
2. Say you had to schedule an appointment with a colleague (if you're a professional) or a professor (if you're a student). How would you do it?
3. How frequently do you check your email, on average, on working days?
4. What languages do you send emails in? Enter as many as you like.
5. Do you have email notifications enabled on your phone?
6. What service do you use for your primary email account?
7. What do you use email for? Select as many as you like.
8. What calendar service do you use, if any? Select as many as you like.

The second part asks participants about using personal assistants for scheduling:

9. If you had a personal assistant, would you ask them to schedule your appointments for you?
10. Imagine that you had an AI-powered, virtual personal assistant who could set up appointments for you by finding available times in your calendar and sending emails on your behalf. Would you use this service?
11. Say that you ask your virtual assistant to schedule an appointment with someone. For the recipient, this email would be indistinguishable from an email sent by a human assistant. Should the virtual assistant inform the recipient that it is not a human, but an AI-powered virtual assistant?
12. Should recipients be able to unsubscribe from your assistant's emails, i.e., "opt out" of talking to an AI, and choose to only communicate with you instead?
13. Would you like to know whether or not someone has seen an email sent by the assistant?

14. If you have enabled email tracking, should end users know that a read receipt has been shared, i.e., a notice at the end of the email, "The sender has been notified that you've opened this email"?
15. Apart from scheduling appointments, what else would you like your virtual assistant to do, over email?

3.2.2 Product Usage

After finishing the Introductory Survey, participants start using the assistant service with the help of virtual guides consisting of several screenshots, an onboarding wizard, and an introductory tour.

The participants open the web application on `https://myeiva.com` and create an account. After creating their account, they verify their email and log into their accounts. They then complete the onboarding flow, setting up basic information such as their name, timezone, gender, security preferences, and name of their assistant.

Then, they follow an introductory tour which helps them customize their assistant's settings, add additional meeting locations, etc., before sending an email to their assistants. After sending an email to schedule an appointment, participants go back to the survey for the UX evaluation.

3.2.3 User Experience Evaluation

In the last part, participants answer questions about the experience of both the assistant and the web application in the form of another survey:

1. What comes to mind when you think about EIVA (how would you describe it to a friend)?
2. On a scale from 1 to 5, review the following about the web app. 1 is the worst and 5 is the best:
   (a) Overall experience of the web app (not the assistant)
   (b) Design of the web app
   (c) Functionality of the web app
   (d) Ease of use of the web app
   (e) Privacy features available in the web app
   (f) Overall experience of the onboarding flow
   (g) Overall experience of the introductory tour
3. On a scale from 1 to 5, review the following about the assistant. 1 is the worst and 5 is the best:
   (a) Overall experience of the assistant over email
   (b) Customizability of the assistant

   (c) Flexibility in understanding natural language
   (d) How human-like does the assistant sound over email? 1 is not human-like at all and 5 is indistinguishable from a human.
   (e) How much do you trust the assistant to not make mistakes? 1 is the least trust and 5 is the most.
   (f) How accurate, practical, and logical were the time recommendations made by the assistant?

4. How many emails did you send with the assistant in CC?
5. Did the assistant understand your email(s) correctly?
6. Did the assistant respond to your email in a reasonable amount of time?
7. Did the assistant recommend the correct location for the appointment?
8. Did the assistant meet your expectations in terms of scheduling an appointment?
9. Which of these features of the web app did you use?
10. If you were the recipient of an email from EIVA, would you be able to tell whether it's an email from an AI or human assistant, if you didn't know?
11. If we launch EIVA as a service where users can get their own assistants, will you use it?
12. Based on the amount of time EIVA saves when setting appointments, how much would you be willing to pay for such a service?
13. Did you find anything frustrating that you wish was easier or different?
14. Is there anything that you wish the assistant could do, in terms of scheduling, that it doesn't currently?
15. What do you like the most about EIVA?
16. What do you like the least about EIVA?
17. Do you have any additional feedback, suggestions, or comments?

3.3 Methods of Analysis

First, the data gathered was prepared for analysis. After closing participation, data received in Google Forms was duplicated in a spreadsheet for analysis. Similarly, Kibana, an open source data visualization dashboard for Elasticsearch, was used to analyze logs for pageviews, API requests, and clicks. For qualitative answers such as long text responses, content analysis was conducted by finding common keywords.

Multiple-choice responses from the Introductory Survey were tabulated. Product usage data collected from logs was visualized in several ways. For one, users' approximate locations were mapped on the maps of the Netherlands, Europe, and the world. Other data such as popular languages, browsers, and operating systems was quantitatively tabulated. Time-series data such as number of API requests, number of emails sent to the assistant, and the website's response time was graphed. Lastly, heatmaps were manually generated using coordinate data from mouse clicks. The mean, median, mode, and standard deviation of rating scale responses from the User Experience Survey were calculated and tabulated.

4 Ideation

The Ideation phase is divided into three parts — Requirements Capture, Assistant Architecture, and Web App Interface. In the first part, a stakeholder analysis is conducted and user personas are used to better understand the usage context of users. In the second part, an early architectural exploration of the email process is conducted. In the last part, mockups and prototypes of the user interface (UI) of the web app are designed and evaluated.

4.1 Requirements Capture

4.1.1 Stakeholder Analysis

The basic stakeholders are the users who will interact with EIVA — the consumers of the assistant service, and the recipients of emails sent by the assistant. However, both these highly interested people have largely different powers; one controls the behavior of the assistant and the other merely interacts with it.

Therefore, although it's important to consider the inputs of each stakeholder, it can be useful to prioritize stakeholders based on how much interest they have in the product, and how much power they have over the designers and engineers building the product. The Power-Interest Grid can be a useful tool to prioritize your tasks based on four categories [34].

1. **High power, highly interested people** (Manage Closely): fully engage these people and make the greatest efforts to satisfy them:
   (a) Users who are using EIVA and have their own assistant that schedules appointments
   (b) External client who is funding and invested in this research project
2. **High power, less interested people** (Keep Satisfied): put enough work in with these people to keep them satisfied, but not so much that they become bored with the message
   (a) Human resources staff at companies who notice a difference in the work of human assistants
3. **Low power, highly interested people** (Keep Informed): adequately inform these people, and talk to them to ensure that no major issues arise
   (a) Users who are recipients in emails sent by EIVA; they don't have the power to control or customize the assistant, but are directly communicating with it
   (b) Human personal assistants whose employment may feel threatened
   (c) People who have signed up to start using the EIVA service on launch
4. **Low power, less interested people** (Monitor): again, monitor these people, but don't bore them with excessive communication

(a) People who have participated in the research and are unsure whether they would want to use the service in the future

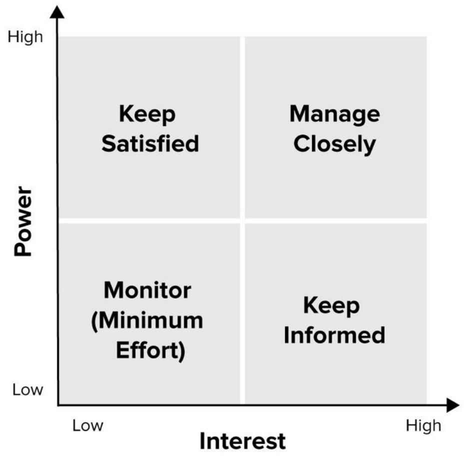


Figure 1: Power-Interest Grid

4.1.2 User Personas

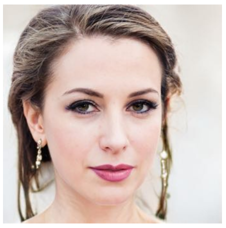
Once basic stakeholders are identified, user personas are used to personify different types of users and study their use cases. User personas are used in user experience (UX) research and help with building empathy for target users. In the ideation phase, personas are used to answer the following questions:

1. Who is the ideal consumer for EIVA?
2. What are current behaviors of those consumers?
3. What are their needs and goals?
4. What are the pain-points that they currently experience?

For this research, four kinds of users are explored in different contexts based on conversations with a diverse group of consumers. To avoid any bias, the Random User Generator service has been used to programmatically generate names of each persona — Tanya Gonzalez, Julia Morris, Joy Fields, and Lee Henderson [35]. For the personas' profile pictures, another online service, This Person Does Not Exist, is used that provides a collection of AI-generated photos from various publicly available research papers and media sources [36].

Tanya Gonzalez Tanya is a 24-year-old MSc student at a 4TU university in the Netherlands and works part-time as a student assistant. Her user environment is the classroom or library, where she spends most of her time. She enjoys trying authentic local cuisine, learning new languages, and reading Harry Potter books.

She has several professors that she wants to schedule appointments with for classes and her assistantship. Her end goal is to have a simple way to view her professors' calendars and find available time slots for meetings. Currently, this takes several email exchanges with each person and wastes a lot of time.



Julia Morris Julia is a 33-year-old freelance makeup artist based in Sydney, Australia. She is in a low-to-median income bracket and earns around AU$45,000. She uses a point of sale (PoS) system in her makeup studio and also uses an iPad when showing her portfolio to prospective clients. She enjoys traveling and her dream is to one day release her own brand of cosmetics.

To keep track of appointments, she uses the default calendar application installed on her iPad. She currently uses WhatsApp to communicate with clients, but her end end goal is to allow them to automatically add bookings to her calendar based on her availability.

Joy Fields Joy is a 47-year-old C-level business executive at a Chicago-based multinational corporation in the financial space. She is in a very high income bracket and earns over US$1MM per year. At work, she mostly uses a MacBook Pro, but also has email and presentation setup on her iPhone because she extensively travels as part of her job. She cares about gender equality, climate change, and data privacy. Most of her time is spent on answering emails and in meetings, for which she uses her company-provided exchange server. She has a full-time assistant, but thinks that his time would be best spent on more productive tasks.

Lee Henderson Lee is a 50 year-old startup founder who previously served in the Air Force. With his new business, he has frequent meetings with partners and investors. His strength is in radio and navigation hardware technology, but still prefers to schedule appointments using pen and paper, a habit he has had since his days in the armed forces. He is not very keen on moving to a digital system, but considers himself open-minded and likes trying new products.

Table 2: Comparison of User Personas

User	Income	Current Behavior	Requirement
Tanya	Low	Email	Appointments with professors
Julia	Medium	WhatsApp	Managing time slots
Joy	High	Personal Assistant	Calls with team
Lee	Medium	Pen-and-paper	Meetings with partners

Though the use cases for each persona are different, they all fall under the umbrella of "I want a better, more automated way of scheduling appointments." For Tanya, it is to interact with professors; for Julia, with clients; for Joy, with her team; and for Lee, with his business partners. With an automated virtual agent that can schedule appointments without input from users, a significant part of all of their problems may be solved.

4.2 Assistant Architecture

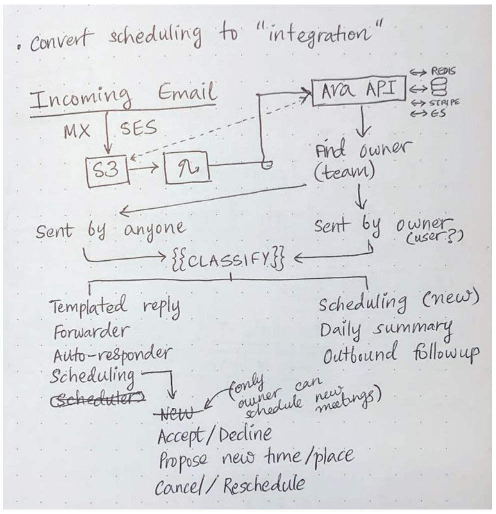


Figure 2: Email Process Ideation

A pen-and-paper exploration of the entire process, from receiving a new email, understanding it, and performing an action, was conducted. In Figure 2, each intent (such as scheduling an appointment, forwarding an email, etc.), is performed based on the classification of the incoming email.

Here, the process of receiving an email, saving the contents, and invoking a serverless function is visualized. The diagram also showcases the connection of the API with other services like a database and payment gateway.

The API then finds the owner of the team the email was addressed to and whether or not the owner themselves sent the email. After classification, one of several processes are queued.

In a separate exploration, a rule-based approach was chosen instead of intent-based classification to understand an incoming email. In Figure 3, a user interface was mocked up wherein consumers could set custom rules to perform actions when new emails are received. In the example, there are three types of UI elements — select dropdowns, text inputs, and buttons.

In the example UI, the rule is ”When you get an email from *anyone*, *and* its *subject contains* the text (input), *and* its *body length is greater than or equal to* (length), *and* the *day is not* one of the days *Sat, Sun*, automatically reply with *template Vacation responder* and also email *me* with the (text). Complicated rules such as these can be generated by the proposed UI, allowing users to automate several different types of repetitive email tasks.

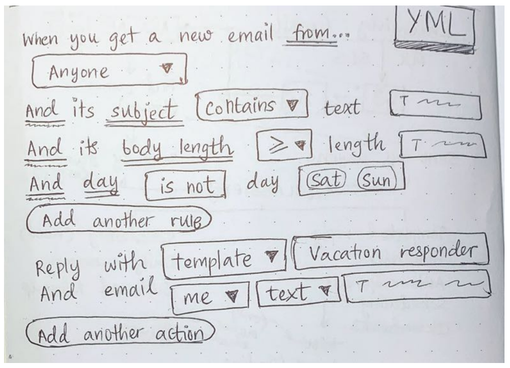


Figure 3: Rule-based Process Ideation

4.3 Web App UI

For setting preferences and managing scheduled appointments, a functional and easy-to-use companion web application must be developed. In Figure 4, early wireframes of the web app are designed, using both desktop and smartphone form factors. After creating an account and verifying their email, users can log in and will be taken to the onboarding interface to set up their assistant. Then, users are taken to the Dashboard page with an overview of their meetings, and can choose to go to the Settings page to configure their preferences.

Figure 4 also shows a two-column layout for desktop pages, with the left sidebar for the primary navigation and a card for the main content. On mobile, a hamburger menu is used with the same design. An accessibility options floating button is also shown on the bottom-right corner of each design. In Figure 5 and 6, more detailed wireframes are drawn using a mobile stencil on paper.

A basic version of each required interface is shown in Figure 5, with the following interfaces from left to right and top to bottom:

1. How to Use: Help new users understand how to communicate with their assistant over email
2. Scheduling: Set scheduling preferences like active days and working hours
3. Login: Sign in to an account using email and password or Google
4. Register: Create a new account with name, email, and password
5. Extensions: Advanced integrations like custom API keys and authentication for Clearbit or Google Calendar
6. Domains: Manage verified domain names for team accounts
7. 2FA: Post-login verification for two-factor authentication

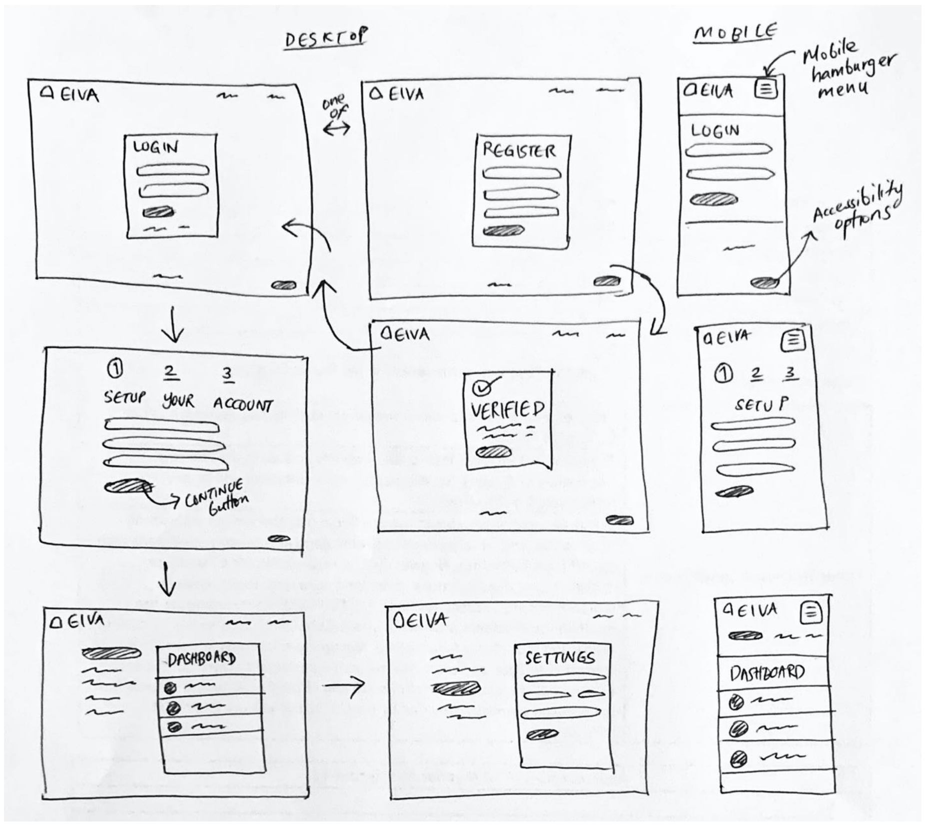


Figure 4: Webpage Wireframes on Paper

8. Face ID: Optional login verification using Apple's Face ID
9. Edit Location: Map interface to update location details
10. Calendar: Overview interface for meetings in the coming month
11. Meetings: A list of upcoming meetings with user avatars
12. Meetings: An expanded card interface for upcoming meetings
13. Calendar: Connections with various calendaring services
14. Calendar: List-view for upcoming appointments
15. Payments: Invoices and credit card details for SaaS payments
16. Locations: List of saved locations with map interface

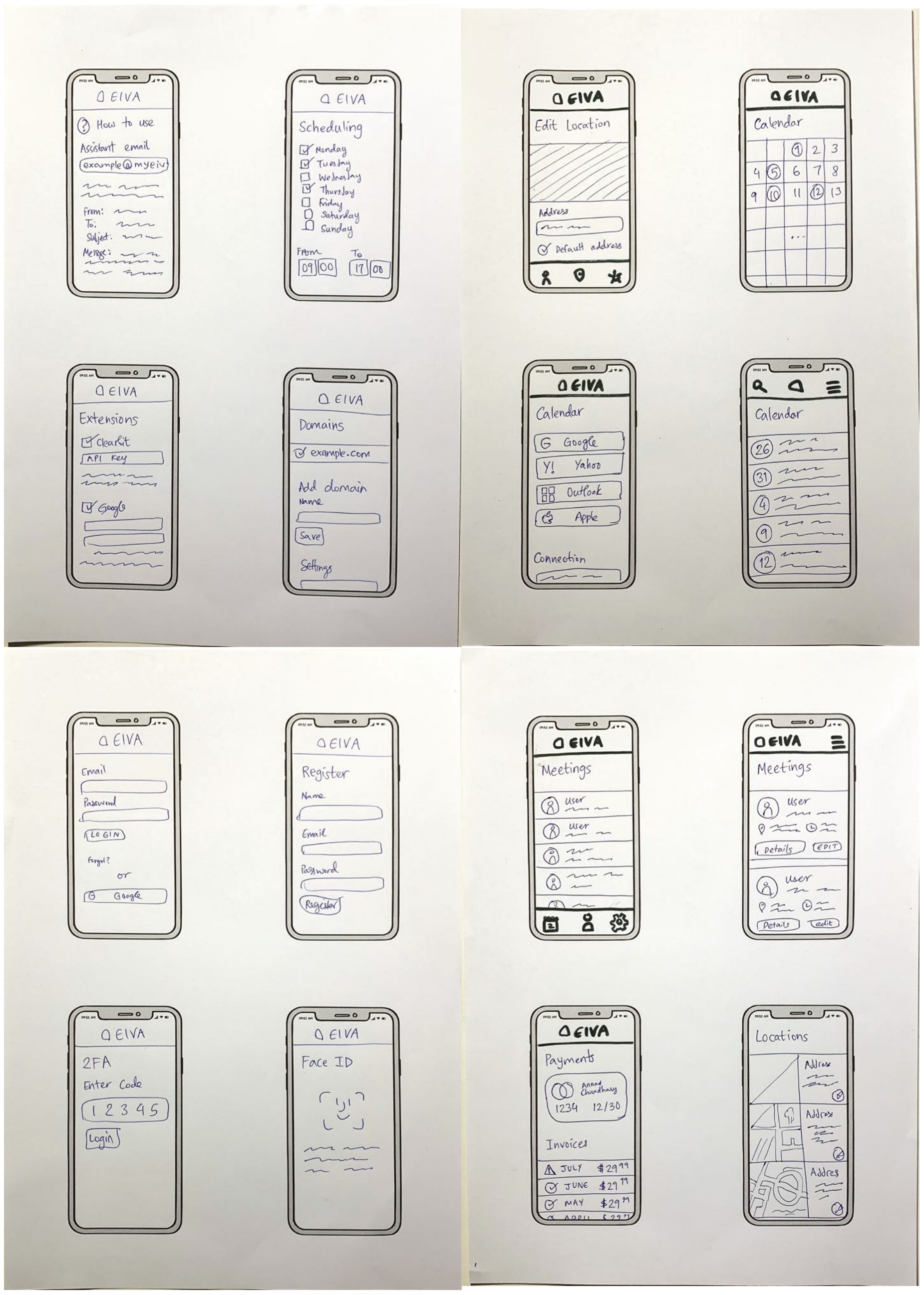
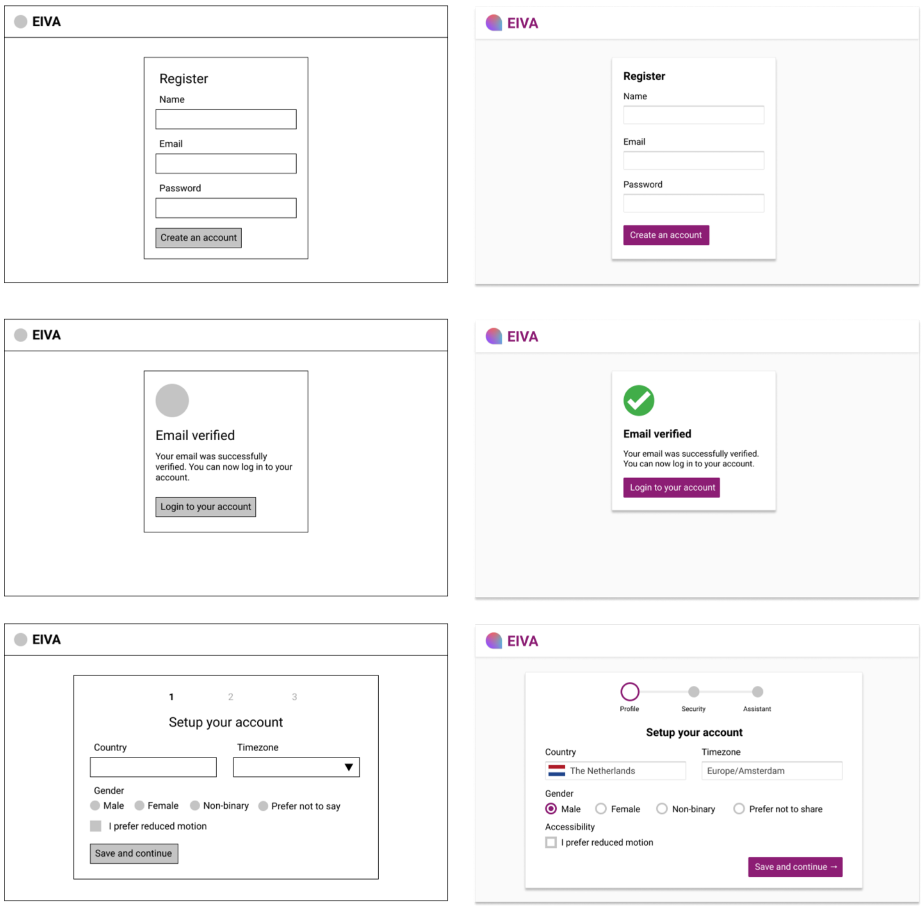


Figure 5: Mobile Wireframes on Paper

Lastly, both low- and high-fidelity mockups of the desktop interface were designed using Figma, a web-based vector graphics editor and prototyping tool [37]. Figure 7 shows a comparison of a selection of these mockups. The low-fidelity mockups are used for layouting and positioning, while the high-fidelity mockups add detail such as colors, shadows, and detailed UI elements like form controls.

Figure 6: Mobile Wireframes Exploration



Your email was successfully verified. You can now log in to your account.

Figure 7: Comparison of low- and high-fidelity mockups

5 Specification

Based on the literature research in Section 2, this chapter focuses on specifying the key requirements for building EIVA. In the first part, the tasks required while scheduling an appointment are understood; and in the second part, personalization and customization aspects are specified.

5.1 Tasks Required in Scheduling by EIVA

Today, most users prefer to use email for scheduling rather than currently available specialized software. A survey of workers in the field of information management found that over 80% of the respondents use email for scheduling and organization [38]. However, this wastes a lot of time because the process of scheduling online appointments is non-trivial and requires several rounds of communication, coordination, and negotiation. This section specifies the required actions for scheduling such appointments.

5.1.1 Understanding natural language over email

Communication with devices using natural language is commonplace for many people today, especially with the advent of affordable smart voice assistants, such as Alexa, available on Amazon Echo; and Google Assistant, available on Google Home [39]. Their interface, which translates a human's intention into a device's control commands using speech recognition and natural language interpretation, is known as a Natural Language Interface (NLI). Some NLIs are extremely useful not only because of the dialogue-style nature of interactions, but also the ability to preserve context across different queries [40]. Therefore, EIVA must remember a user's preferences from previous meetings.

Personification of the scheduling process using natural language has the additional benefit of increased likeliness of error forgiveness by end users. In a study, over 80% of the participants who encountered errors in NLIs understanding them said that either their attitude was unaffected or the users themselves "must have typed in something wrong" [41]. Furthermore, NLIs allow people to use their existing calendaring tools and idiosyncrasies by adopting email as the user interface and natural language conversations as the interaction mode. For example, a message such as "Please schedule an appointment with Florian for next week" should be enough for EIVA to take over the remainder of the process.

5.1.2 Parameter recommendations

Recommendation systems are frequently used in several industries to help users make better decisions. Examples of collaborative recommender systems include product recommendations on e-commerce website Amazon [42] and TV series recommendations on the streaming provider Netflix [43]. Modern systems may use various machine learning techniques like artificial neural networks combined with feature extraction methods to find the ideal recommendation [44].

Scheduling a single appointment requires agreement on several parameters, such as date, time, duration, and location of meeting. Recommending a set of these parameters must be based on the user's predetermined preferences, such as the hours they prefer for scheduling calls. However, the ideal implementation of such a recommendation system may take a multidimensional approach that uses additional contextual information [45]. For example, contextual information may include the user's location to account for the travel time to a recommended meeting's location; or, for international calls, it may include the guest's timezone based on their IP address.

5.1.3 Negotiation

Since not all guests may be available at the initially proposed time, the next major task required by EIVA is negotiation. The negotiation mechanism allows it to be flexible about constraints imposed by the preferences of other users. Zunino and Campo (2009) note that the negotiation mechanism "starts when two [users] do not agree on some parameter of a meeting," such as the date, time, or location of the meeting [46]. Then, the guest may propose a new set of parameters for the meeting, for example by modifying its time. EIVA will evaluate whether the proposal is convenient or not by comparing the "likeliness of reaching a consensus using the previous meeting parameters with the proposed parameters." In this step, it can also make a judgement call by choosing to ignore part of the user's preferred parameters in order to agree with the guests. This back-and-forth negotiation process needs to occur asynchronously, "sometimes requiring several days for the parties to reach consensus" [8].

5.1.4 Renegotiation and confirmation invitations

In the dynamic environment that is the modern workplace, changing appointment parameters is commonplace. A study found that 66% of all scheduled meetings by a CEO underwent a parameter change [7]. Inevitably, once a meeting is scheduled, it requires "continuous maintenance, as new events often prompt meeting updates and reschedules" [8]. To finalize the meeting parameters, EIVA should take into account the parameter preferences of both the user and all guests.

To make sure that all guests have the most recent parameters, confirmations and reminders may be sent by EIVA. However, not all users use the same calendaring software, with some professionals using no digital calendar at all. Kincaid reported that approximately one half of all meetings scheduled by professionals were with users using a different calendaring system than their own [19]. Therefore, EIVA will be required to send these emails using industry standards supported by most software. The primary standard to store and exchange calendaring and scheduling information is the Internet Calendaring and Scheduling Core Object Specification (iCalendar) [47].

5.2 User Configuration

5.2.1 Authorship

One of the first discussed questions with the client is whether or not recipients should clearly know that an email sent by EIVA is not actually sent by a human assistant. This question of authorship is not just a technical one, but an ethical one as well. In journalism, for example, there are articles entirely written by AI-powered virtual agents, which raises the question *Who is the author of an article written by a virtual agent?* A 2005 study found that research participants attribute story credit to the programmers who developed the AI or the news organization publishing the story [48].

The key reason why authorship is important is credibility. If you know the name of the author of an article in a major newspaper or magazine, you can find out more information about them, perhaps by visiting their social media handles. If you have any questions about their work, or found a mistake in their article, you can contact them directly. In the case of an article written by an unnamed virtual agent, the only option is to find the contact information of the publication. This is also the case with EIVA, where users can name their own assistant and essentially create a human-like entity, but without the necessary digital history required for authenticity.

Authorship Clarity In news articles written by virtual agents, there is “no visible indicator for readers to verify whether an article was written by a robot or human”, which raises issues of transparency [49]. In the case of EIVA, if an AI assistant is impersonating a human assistant to send emails on the professional’s behalf, it raises the same ethical question of whether the end user receiving the email should know that it was not written by a human. Some users may not wish to disclose this information and consider authorship a “trivial” aspect. While for others, this sets a powerful and potentially harmful precedent for AI authorship and will choose to make that information clear.

Therefore, it is recommended to keep such decisions customizable. Using the EIVA website’s settings page, consumers should be able to set preferences about whether they want their assistant to inform end users about the fact that they are virtual agents, not human assistants. By default, this is preselected for all users, and is a simple checkbox user interface with the message “Inform recipients that your assistant is a virtual assistant”. This clarity highlights the commitment to personalization that such a product should bring.

5.2.2 Sensible Defaults

The previous section sheds light on the importance of defaults, such as letting recipients know about the AI nature of the assistant by default. Using the feedback from users, especially for customizable options related to individual privacy, the less secure and more open options should always be opt-in, not opt-out, to ensure sensible rather than opinionated defaults. Other configuration options can be based on the results of the initial survey with users and feedback from stakeholders.

It has been estimated that 95% of all users stick to the default settings in an application, and don't bother to change them [50]. Therefore, this is an important responsibility on the hands of designers and engineers to ensure that the default options are most secure. In EIVA, options such as email tracking, directly connecting a calendar, or even sharing personal video meeting links, are all optional and opt-in, so the default state is always most secure.

5.2.3 Privacy-first

The privacy precedent that most virtual assistants have set is not highly positive. Personal assistants in the form of smart speakers like Amazon Echo (running Alexa) and Google Home (running Google Assistant) have "begun to significantly alter people's everyday experiences with technology" [51]. These virtual assistants can continually improve with increased usage using deep learning [52]. Although this makes the assistants more useful over time, they also "amplify the overall debate about privacy issues" [53].

For example, an Echo device unintentionally recorded a Portland family's private conversations and shared it with a random person from their contact list [54]. For EIVA, the learning is simple: make sure no personal information is recorded if it is not strictly necessary, and users should be able to configure what they want to share.

5.2.4 Gender Bias

There is often visible gender bias when it comes to personal assistants or secretaries [55]. Historically, these were executive assistants, but modern technologies have translated these biases to virtual assistants as well. When asked about why Cortana is female, a Microsoft spokesperson said Cortana can technically be genderless, but the company did immerse itself in gender research when choosing a voice and weighed the benefits of a male and female voice [56], "for our objectives — building a helpful, supportive, trustworthy assistant — a female voice was the stronger choice".

Apple's Siri and Google Assistant offer the option of changing the voice to male, but Alexa and Cortana don't have male counterparts. The first United Nations' examination of gendering of AI technology found that "gender imbalances in the digital sector can be 'hard-coded' into technology products" [57]. In that report, the following problems are highlighted:

- Digital assistants reflect, reinforce, and spread gender bias
- They model acceptance and tolerance of sexual harassment and verbal abuse
- They send explicit and implicit messages about how women and girls should respond to requests and express themselves

- They make women the ‘face’ of glitches and errors that result from the limitations of hardware and software designed predominantly by men
- They force synthetic ‘female’ voices and personalities to defer questions and commands to higher (and often male) authorities

The UN report makes 15 recommendations, including “[ending] the practice of making digital assistants female by default.” For EIVA, this is an important lesson because with the intention of sounding friendly, the original name for EIVA was Ara. However, the gender role that accompanies this is a big problem, therefore the name was changed to EIVA, which is an acronym for Email-based Intelligent Virtual Assistant, and yet sounds friendly and human-like.

However, though the name of the service is EIVA, as discussed in the previous section, customizability is a key part of the user experience, and users should be able to pick custom names for their assistants via the interface. Unisex names like Alex, Jesse, Urooj, and Daya can be used by end users; the idea is to make the assistant completely customizable. Furthermore, users can set culture-specific names, giving EIVA a wider cultural appear. To make this easy, the first setting in the user interface is to select the assistant’s name and signature, which users can freely pick.

6 Realization

The EIVA service consists of a frontend web app that users directly interact with and a backend API that powers it.

6.1 Backend APIs

EIVA's backend project is based on Staart API, an open source framework to build Software as a Service (SaaS) products with TypeScript. With Staart API, several features that typical web apps require, such as user management, organizations, authentication, API gateway, etc., are built-in and do not require significant effort for setup. Staart API is also built by the author of this graduation project and has over 200 stars on GitHub. Figure 8 shows the eight services that the EIVA API requires, including third-party services like Clearbit and Stripe, AWS services like SES and S3, and self-hosted services like Redis.

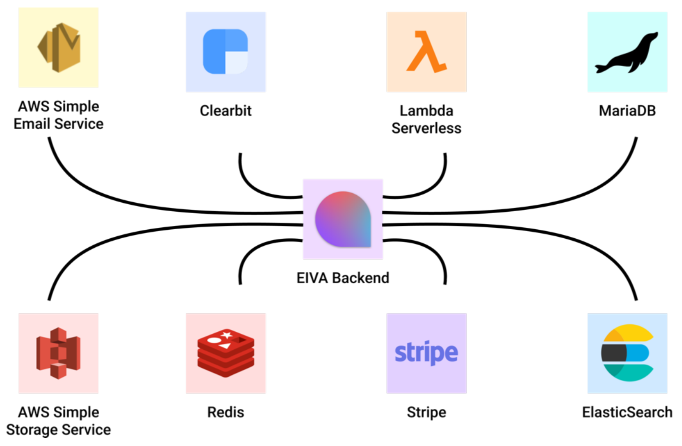


Figure 8: EIVA API Connections

6.1.1 Configuration Management

Using ES Modules, the ECMAScript standard for working with modules, data and structure types can be imported and exported [58]. The `src/config.ts` file acts as the central configuration manager and sets default values for keys not available in the environment.

The Twelve-Factor App methodology recommends storing configuration in the environment, separate from the code, and `dotenv` is used to read environment variables from a `.env` file [59]. This library makes values available in `process.env`, Node.js's object to access environment variables.

For example, the `PORT` constant defines the port the app should listen on, and defaults to port 80:

```
import { config } from "dotenv";
config();
export const PORT = process.env.PORT ? parseInt(process.env.PORT) : 80;
```

Then, when initializing the app, this value is imported:

```
import { PORT } from "./constants";
import { Server } from "@staart/server";
export class Staart extends Server {
  public start(port: number = PORT): void {
    this.app.listen(port, () => console.log(`Listening on ${port}`));
  }
}
```

This central management makes it very easy to update defaults (like port 80), and the `.env` file contains environment-specific values, like port 7000 for development and 443 for production. A `.env.example` file is available in the repository for reproduction. A total of 68 exported members configure various parts of the application, from database connection URLs to secret keys for signing JSON Web Tokens (Section 6.1.3).

6.1.2 Natural Language Processing

There are three Natural Language Processing (NLP) systems implemented in EIVA's backend — language detection and variable extraction, intent classification, and date parsing. These systems work together during the scheduling process (Section 6.1.9) and use various libraries.

Language Detection and Variable Extraction To identify the language of an incoming email, Google Cloud's Natural Language API is used. This provides both the language and list of entities found, such us names of people and places. Using this API is free of cost for up to 5,000 requests every month, which is more than enough for this project, since this API is only used to identify the language to ensure the response of the email is also in the correct language.

Intent Classification A Naive Bayes classifier is used to identify the intent of an incoming email from one of four categories — setting up a new appointment, rescheduling an existing appointment, canceling an appointment, or sending a summary of the appointments for the day.

Naive Bayes classifiers are probabilistic classifiers, meaning that they calculate a probability distribution over a set of classes, and are based on applying Bayes' theorem with strong independence assumptions between the features. Features are independent if the realization of one does not affect the probability distribution of the other; for example, a fruit can be considered an apple if it is round and red, and the naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color or roundness [60].

The `BayesClassifier` class of Natural, a general natural language facility for Node.js, is used for the naive Bayes classification. The training data consists of common phrases, for example:

```
import { BayesClassifier } from "natural";
const classifier = new BayesClassifier();

classifier.addDocument(
  [
    "set up an appointment",
    "schedule a call",
    "meet me for dinner",
  ],
  "setupNewAppointment"
);
classifier.addDocument(
  [
    "i can't make it",
    "reschedule this call",
    "find another time",
  ],
  "rescheduleAppointment"
);

classifier.train();
```

In the above example, when the classifier receives a new string, such as "I don't think I'll be able to make it this time", it will return a higher probability for `rescheduleAppointment` than `setupNewAppointment`, as the following example shows:

```
classifier.getClassifications(
  "i don't think I'll be able to make it this time"
);
```

Here, the value of rescheduling is 0.333... and that of setting up a new appointment is 0.1666..., which shows that the naive Bayes classifier works at a decent accuracy, even with an extremely small training dataset. In EIVA, the training dataset is larger and there are four categories of classifications.

Date Parsing Lastly, Chrono, a date parsing library, is used to understand natural language dates and times. Chrono can understand specific days like "today" and "tomorrow", relative dates and times like "this Friday at 13:00" and "two weeks from now", and absolute values like "Aug 17, 2020".

In the Netherlands, the convention is to write times in 24-hour format with a period as the hour-minute separator (e.g., 14.30 instead of 2:30 pm) [61]. Since Chrono is not optimized for understanding this format, a regular expression (regex)-based replacement from period to colon is done on the entire text string before parsing. Other special cases, such as no separator, are also handled.

6.1.3 Transactional Emails

EIVA does not send any promotional emails, only transactional emails for authentication-related cases like email verification and password reset links, and communication to the guests or host from the assistant [62].

Email templates are defined in Markdown, and rendered with variables using Mustache. Markdown is a lightweight markup language with plain-text-formatting syntax created by John Gruber with Aaron Swartz, and was selected because of its ease of conversion to both HTML and plain text [63]. Since there are also several variables that need to be replaced in the template, Mustache, a web template system is used alongside Markdown.

The author's `@staart/mustache-markdown` package makes rendering easy by exposing a single interface for both Mustache and Markdown [64]. For example, an email verification email template may contain:

```
Hello, {{name}}! Please click on this link to verify your email:
![{{link}}](Verify my email).
```

When an object containing data is provided (say, the name 'Florian' and an example link), the following HTML is generated:

```
<p>Hello, Florian! Please click on this link to verify your email:
<a href="https://example.com">Verify my email</a>.</p>
```

Emails are sent using Amazon Web Services' Simple Email Service (SES) using Nodemailer, a library that provides a simple API to send email using Node.js and supports both SES and SMTP [65]. Each email is sent with both the plain text and HTML versions, with DKIM and SPF verification.

For "magic links" for email verification, password reset, or appointment confirmation, JSON Web Tokens (JWT) are used. JWTs are URL-safe and have encrypted data, and can be verified server-side without additional database queries. A queue is used to send emails sequentially, implemented using Redis (Section 6.1.5). This ensures that the rate limits imposed by SES are respected, and that undelivered emails can be sent again (up to 3 tries).

6.1.4 Authentication

Users can create an account using the web app by entering their name, email, and password. Passwords are hashed using `bcrypt` before storage, which introduces a work factor and is therefore much safer than general-purpose hash functions [66]. It also has built-in salting to prevent rainbow table attacks [67].

For authentication, the industry-standard access/refresh token combination is used. When logging in, both tokens are generated and stored on the client (Section 6.2.1), with the access token expiring after 15 minutes and the refresh token after 30 days. The expired access token can be exchanged for a new one using the refresh token.

Both tokens are JWTs and contain all required information; the access token includes details about the user (such as their name and role) whereas the refresh token only includes the user ID. Furthermore, the web app also allows users to see a list of their logged in sessions and invalidate refresh tokens, which are then blacklisted in Redis.

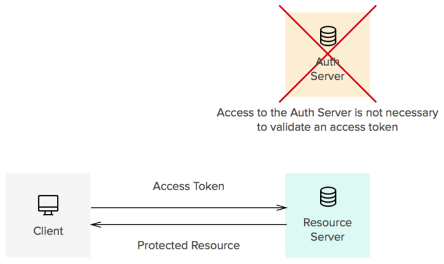

Auth Server
Access to the Auth Server is not necessary to validate an access token
Access Token
Client
Protected Resource
Resource Server

Figure 9: Access Token Process [1]

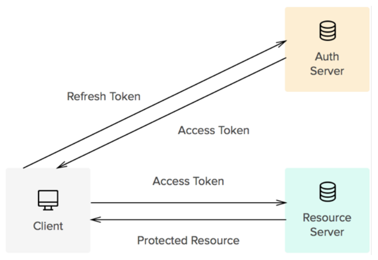


Figure 10: Refresh Token Process [1]

Since many professionals prefer using multiple emails (perhaps a personal and a work email) and the assistant should recognize all their emails, a user account can be associated with multiple emails. This also means that users are free to log in with any of their verified emails.

Rate Limits Rate limits are used to control the rate of requests to the API and is also used to prevent DDoS attacks [68]. An HTTP 429 (Too Many Requests) error is thrown if a client makes too many requests within a given time frame. By default, 60 requests per minute are permitted per IP address.

For programmatic access to the service, users can generate API keys or access tokens to control their assistant or accounts respectively. The frontend web app also uses such an API key for an increased rate limit of 1,000 requests per minute. An in-memory key-value store is used to track these limits.

Lastly, brute force and speed limits are also imposed. To prevent brute force attacks on authentication endpoints like login and register, more than 5 requests in a period of 5 minutes results in a delay of 250 milliseconds per request. Similarly, all other endpoints also add a 100 millisecond per request delay after 1,000 requests within ten minutes.

6.1.5 Database and Storage

MariaDB is used as the primary database management system for the backend. MariaDB is a drop-in replacement for and a community-developed fork of MySQL, intended to remain free and open source software under the GNU General Public License [69]. For communication with the database, the open source database toolkit Prisma is used. Prisma replaces object-relational mapping with a query builder and TypeScript schema generator. It also allows for database migrations and versioned schema stored in the repository [70].

Relations are used extensively in the database; for example, the `users` table has a reference to nearly all other tables. Figure 11 visualizes a simplified version of the database schema, and Appendix 5 contains the entire schema.

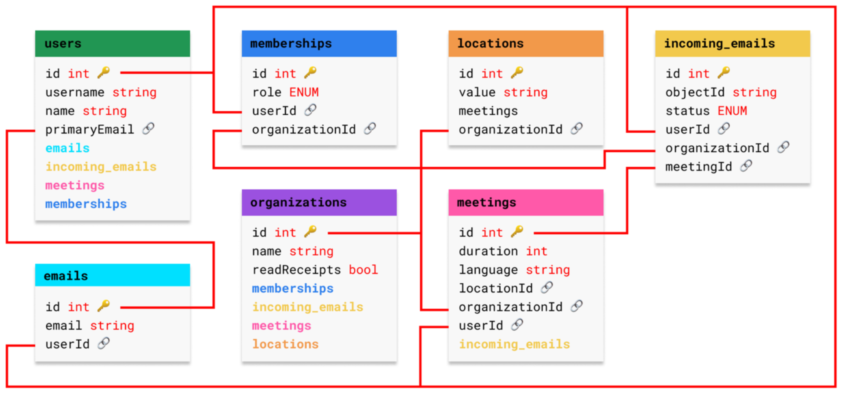


Figure 11: Simplified Database Schema with Relations

Redis Apart from MariaDB, Redis is used for storing data that requires minimal query latency. Redis is an in-memory key–value database, and in-memory processing has been called "the ultimate solution to latency" [71]. For workflows such as checking if a JWT has been invalidated, a MariaDB database query takes hundreds of milliseconds, whereas Redis only takes tens, and is therefore used for caching and invalidation in EIVA.

Redis is also used as a message queue using Redis Simple Message Queue (RSMQ) which implements a queue with guaranteed message delivery to one recipient. This is used to queue outbound webhooks and emails, which are sequentially sent every minute using a scheduled job (Section 6.1.7).

ElasticSearch ElasticSearch, a Lucene-based search engine, is used for storing server logs and analytics data [72]. ElasticSearch is open source and capable of full-text search, which makes it extremely useful to sort through thousands of usage events [73]. Amazon Web Services (AWS) offers Open Distro for ElasticSearch with increased security features, including data encryption at rest and Kibana authentication [74], which is used for this project. In Section 7.2, data stored in this ElasticSearch instance is computed and analyzed.

AWS S3 Although there are no binary files, the API still deals with large text files like raw incoming emails, typically between 1 and 10 kilobytes in size. A database is not ideal for storing such documents, so AWS's Simple Storage

Service (S3) is used. S3 provides object storage along with backup, recovery, and encryption, and provides a Node.js API wrapper to easily upload and fetch files.

6.1.6 API Endpoints

The backend API endpoints are versioned with a `/v1` prefix, and categorized. For example, the login endpoint is `POST /v1/auth/login`. Since there are tens of total endpoints, it's useful to categorize them for development and management; each API category is defined in Table 3. Each endpoint category prefix is automatically generated from the `controllers` directory in Staart API's build process, based on the subdirectory it is located in. For example, the user emails controller file `controllers/users/_id/emails.ts` generates the endpoint `/v1/users/:id/emails` where `:id` is a user ID.

Table 3: API Prefixes

Prefix	Description
`/admin`	Administration and analytics
`/api`	Public API
`/auth`	Authentication
`/organizations`	Teams and assistant settings
`/users`	User settings
`/webhooks`	Incoming webhooks

Public API The internal process that EIVA uses to schedule appointments (Section 6.1.9) has also been designed in a way to be consumable by other clients. A public API has been developed for each action as listed in Table 4.

Table 4: Scheduling API Endpoints

Method	Endpoint	Description
`POST`	`/classify`	Classify text using NLP
`POST`	`/parse-email`	Parse a raw email
`POST`	`/smart-tokenize`	Intelligently tokenize text

Developers can use these endpoints in the future to intelligently understand incoming emails and build other applications on top of the EIVA platform.

6.1.7 Scheduled Jobs

To automate common tasks, specific jobs are scheduled using `cron`, a Node.js API for the time-based job scheduler of the same name. For example, new data is fetched from message queues every minute to send outbound emails, trigger outbound webhooks, and store logs in ElasticSearch.

Table 5: Scheduled Jobs

Expression	Repetition	Job
`0 0 * * *`	Every day	Delete old server logs
`0 * * * *`	Every hour	Delete expired sessions
`0 * * * *`	Every hour	Update errored emails
`* * * * *`	Every minute	Get new data from message queues

6.1.8 Third-party APIs

Other third-party APIs are used for specific utilities, such as Clearbit to fetch details about a person from their email address. Clearbit's Enrichment API uses over 250 public and private sources and provides several datapoints, such as name, profession, company, and timezone. This is used to better understand the context of guests in meetings. Other APIs used in EIVA include Stripe to accept end-user payments [75].

6.1.9 Scheduling Process

To schedule a new appointment, users have to send an email to their guests with their assistant in CC. Every assistant has an auto-generated but customizable `@myeiva.com` email address that is shown on the *How to Use* page (Section 6.2.3). Figure 12 shows that every time a new email is received by AWS SES, the raw email is stored in AWS S3 and a Lambda serverless function is triggered, which in turn triggers an API webhook. The API webhook is triggered with an HMAC-SHA256 token consisting of the object ID of the S3 email signed with a shared secret to ensure that the webhook is not publicly usable.

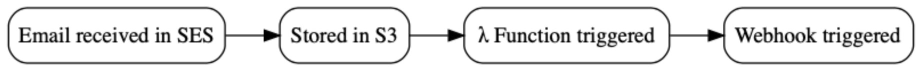


Figure 12: Email Received to Webhook Process

When the webhook is triggered, the API starts with trying to understand the incoming email. Figure 13 graphs the process from trigger to classification, consisting of fetching the raw email from AWS S3, parsing the raw email, finding the team and user based on the assistant's email, setting up the database record, and tokenizing and classifying the email contents.

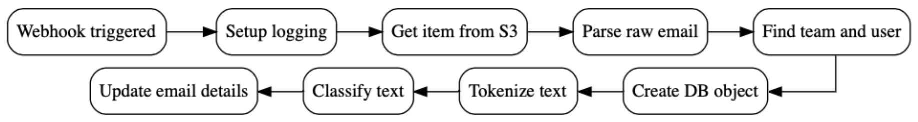


Figure 13: Webhook to Classification Process

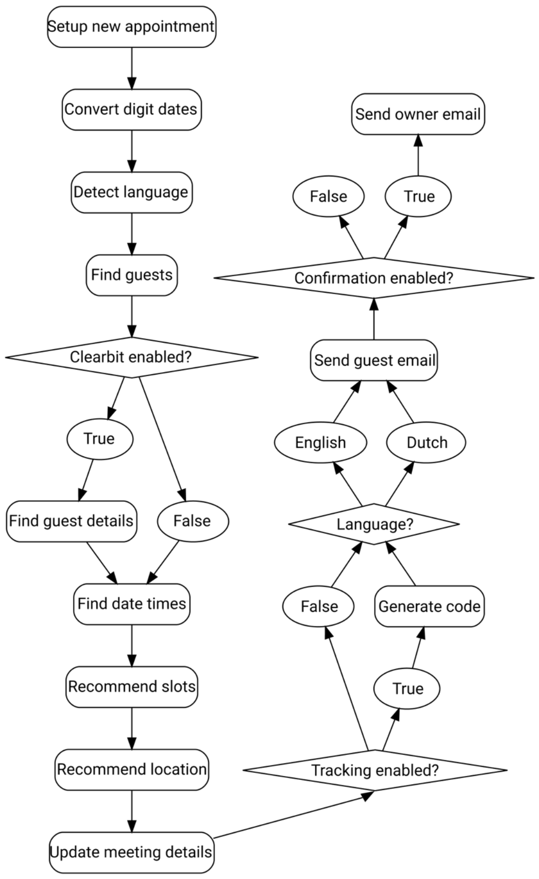
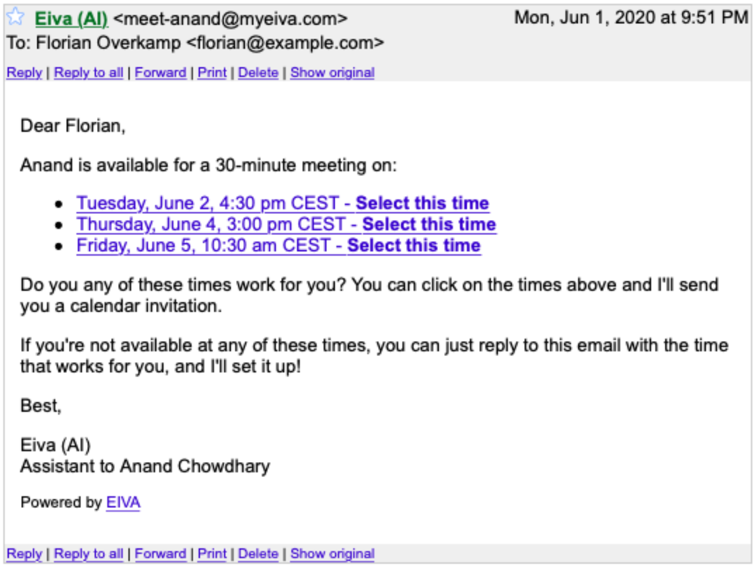


Figure 14: Scheduling Process

After the email is classified as requesting a new appointment, the process described in Figure 13 is triggered to schedule a new appointment for the user and their guests. First, the entire email text is modified for consistent date formats; then, a Google API is used to detect the language of the email (Dutch or English); then, all the guests are detected, and Clearbit is used to find additional information about them, if enabled. Then, dates and times are found using Chrono (Section 6.1.2), and 3 time slots are recommended along with the location and an email is sent. An example email is visible in Figure 14.

After receiving the list of time recommendations from the assistants, guests can click on their preferred option for confirmation (Figure 20), and an email with a calendar invitation is sent to both the host and the guests.


Eiva (AI) <meet-anand@myeiva.com> Mon, Jun 1, 2020 at 9:51 PM
To: Florian Overkamp <florian@example.com>

Reply | Reply to all | Forward | Print | Delete | Show original

Dear Florian,

Anand is available for a 30-minute meeting on:

- Tuesday, June 2, 4:30 pm CEST - **Select this time**
- Thursday, June 4, 3:00 pm CEST - **Select this time**
- Friday, June 5, 10:30 am CEST - **Select this time**

Do you any of these times work for you? You can click on the times above and I'll send you a calendar invitation.

If you're not available at any of these times, you can just reply to this email with the time that works for you, and I'll set it up!

Best,

Eiva (AI)
Assistant to Anand Chowdhary

Powered by EIVA

Reply | Reply to all | Forward | Print | Delete | Show original

Figure 15: Screenshot of Time Recommendation Email

6.2 Frontend Web App

The frontend web app is built with web development best practices, ease of use, and accessibility in mind.

For example, the EIVA homepage receives a 100% score in the Accessibility audit part of Webkit Developer Tools. On Lighthouse, an automated performance auditing tool, its score is 61/100 for performance, 93/100 for accessibility, 100/100 for best practices, and 92/100 for SEO. Oswald Labs' web accessibility plugin is also implemented on every page, which helps people with dyslexia read more easily, amongst other features [76].

6.2.1 Vue.js and Nuxt.js

As described in Section 2.2.2, Vue.js is the frontend development framework of choice for developing the EIVA web app. However, when building large-scale apps, setting up processes for routing, state management, and deploying can take up a lot of time. Therefore, Nuxt.js, an open source framework built on top of Vue.js is used which has these features built-in. For example, Nuxt.js can automatically generate routes based on file names, and also supports TypeScript; it "provides a simple and intuitive developer experience to prototype blazing fast modern web application" [77].

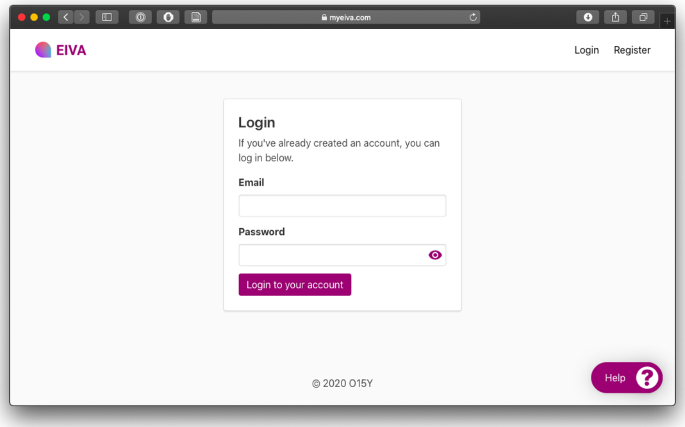
Custom middleware and plugins were created and added to Nuxt.js for features like click tracking (Section 7.2.3), unauthenticated redirects, and persisted state (Section 6.2.1).

Fetch Requests The authentication process grants an access and a refresh token on successful login (Section 6.1.4), which is stored centrally in the web app. HTTP requests to the API have the `Authorization` header consisting of the access token when the user is logged in. Requests are sent using Axios, a Promise-based HTTP client built on top of `XMLHTTPRequest`.

State Management A central store was used for authentication, consisting of both tokens and details about the logged in user, such as their memberships. Vuex, Vue.js's state management pattern and library, is used with types and rulesets ensuring that the state can only be mutated in a predictable fashion. The store is also persisted locally in the browser's local storage as a JSON object, so that authentication details are not lost if the user closes and returns to the web app at a later time.

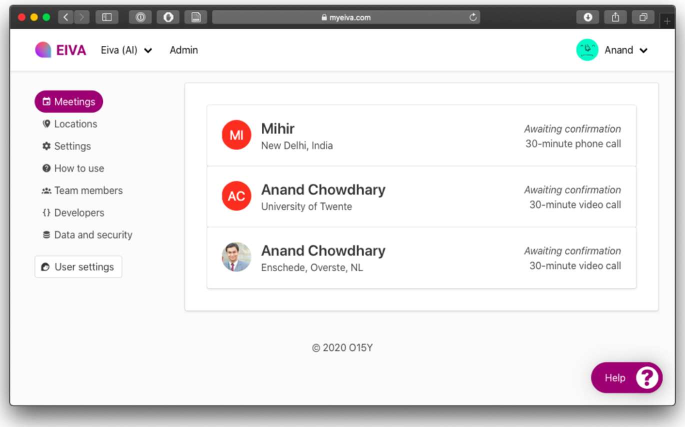
6.2.2 User Interface

The web app UI was wireframed, prototyped, and mocked up extensively before implementation (Section 4.3). It consists of simple branding and a color scheme focusing on desired actions. Figures 16 to 23 are screenshots of important webpages.



If you've already created an account, you can log in below.

Figure 16: Screenshot of Login Page



Mihir
New Delhi, India
Awaiting confirmation
30-minute phone call

Anand Chowdhary
University of Twente
Awaiting confirmation
30-minute video call

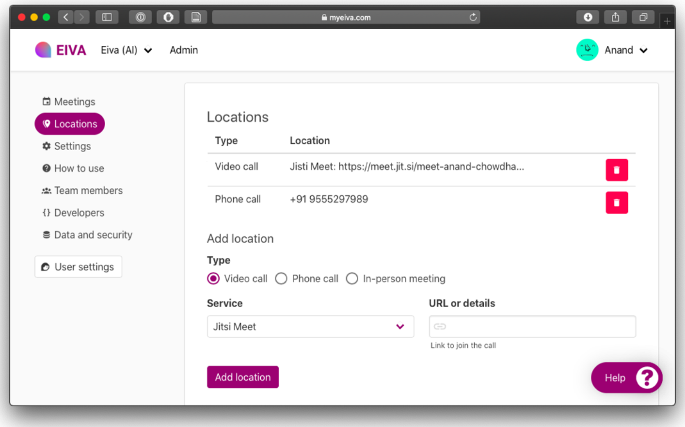
Anand Chowdhary
Enschede, Overste, NL
Awaiting confirmation
30-minute video call

Figure 17: Screenshot of Meetings Page

6.3 Product Development Cycle

During development, a simplified version of the Software Development Life Cycle and the New Product Development Process were adopted for this project, with the following steps:

1. Idea



Locations

Type	Location
Video call	Jisti Meet: https://meet.jit.si/meet-anand-chowdha...
Phone call	+91 9555297989

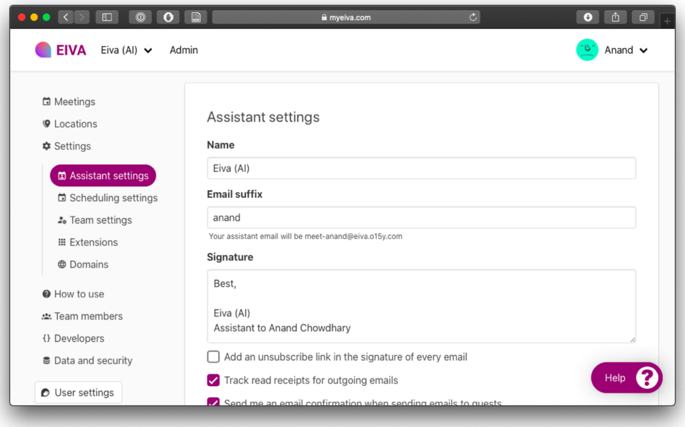
Add location

Type
[✓ Video call] Phone call In-person meeting

Service: Jitsi Meet

URL or details:
Link to join the call

Figure 18: Screenshot of Locations Page



Assistant settings

Name: Eiva (AI)

Email suffix: anand

Your assistant email will be meet-anand@eiva.o15y.com

Signature:
Best,

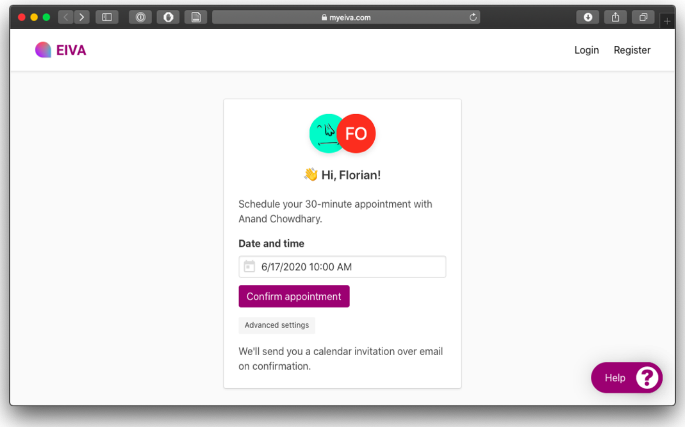
Eiva (AI)
Assistant to Anand Chowdhary

☐ Add an unsubscribe link in the signature of every email

☑ Track read receipts for outgoing emails

☑ Send me an email confirmation when sending emails to guests

Figure 19: Screenshot of Assistant Settings



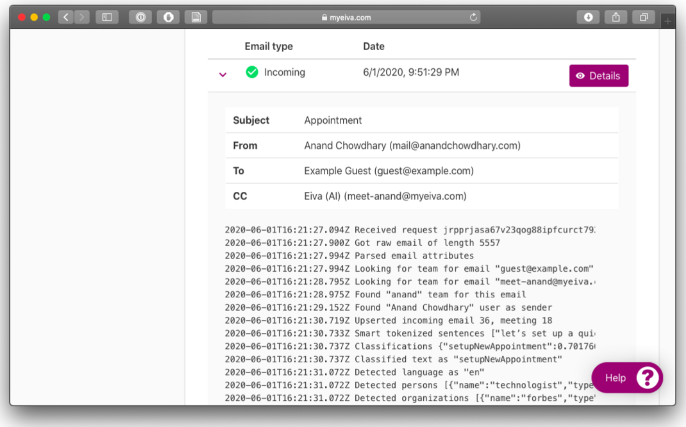
👋 Hi, Florian!

Schedule your 30-minute appointment with Anand Chowdhary.

Date and time: 6/17/2020 10:00 AM

We'll send you a calendar invitation over email on confirmation.

Figure 20: Screenshot of Confirmation Page



Email type	Date
Incoming	6/1/2020, 9:51:29 PM

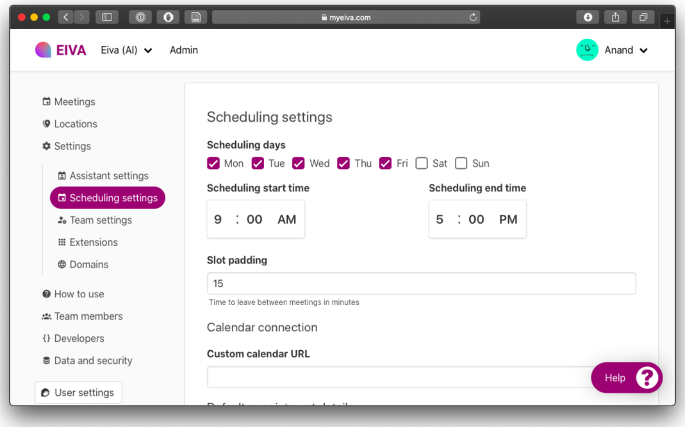
Subject	Appointment
From	Anand Chowdhary (mail@anandchowdhary.com)
To	Example Guest (guest@example.com)
CC	Eiva (AI) (meet-anand@myeiva.com)

```
2020-06-01T16:21:27.094Z Received request jrpprjasa67v23qog88ipfcurct79
2020-06-01T16:21:27.900Z Got raw email of length 5557
2020-06-01T16:21:27.994Z Parsed email attributes
2020-06-01T16:21:27.994Z Looking for team for email "guest@example.com"
2020-06-01T16:21:28.795Z Looking for team for email "meet-anand@myeiva.
2020-06-01T16:21:28.975Z Found "anand" team for this email
2020-06-01T16:21:29.152Z Found "Anand Chowdhary" user as sender
2020-06-01T16:21:30.719Z Upserted incoming email 36, meeting 18
2020-06-01T16:21:30.733Z Smart tokenized sentences ["let's set up a qui
2020-06-01T16:21:30.737Z Classifications {"setupNewAppointment":0.70176
2020-06-01T16:21:30.737Z Classified text as "setupNewAppointment"
2020-06-01T16:21:31.072Z Detected language as "en"
2020-06-01T16:21:31.072Z Detected persons [{"name":"technologist","type
2020-06-01T16:21:31.072Z Detected organizations [{"name":"forbes","type'
```

Figure 21: Screenshot of Incoming Email Logs Page

2. Research
3. Development
4. External testing
5. Analysis

The first step builds on top of the ideation phase with methodological exploration of the idea and specifying the required feature as a module. In the second step, the technical literature research presented in Section 2 is studied to architect the module's technology. In the third step, the idea is materialized



Scheduling settings

Scheduling days

[✓ Mon] [✓ Tue] [✓ Wed] [✓ Thu] [✓ Fri] Sat Sun

Scheduling start time	**Scheduling end time**
9 : 00 AM	5 : 00 PM

Slot padding

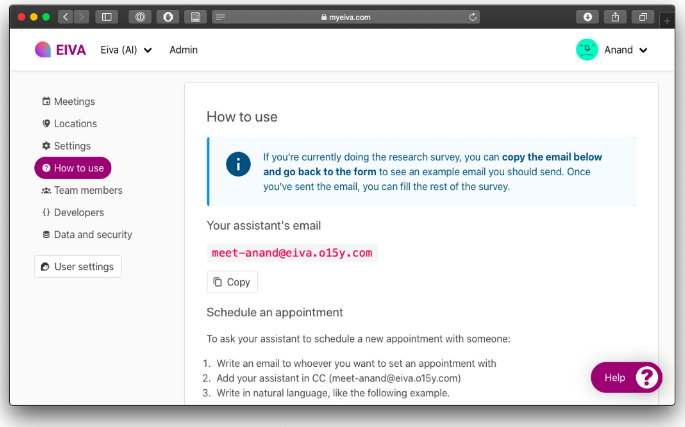
15

Time to leave between meetings in minutes

Calendar connection

Custom calendar URL

Figure 22: Screenshot of Scheduling Settings



How to use

If you're currently doing the research survey, you can **copy the email below and go back to the form** to see an example email you should send. Once you've sent the email, you can fill the rest of the survey.

Your assistant's email

`meet-anand@eiva.o15y.com`

Schedule an appointment

To ask your assistant to schedule a new appointment with someone:

1. Write an email to whoever you want to set an appointment with
2. Add your assistant in CC (meet-anand@eiva.o15y.com)
3. Write in natural language, like the following example.

Figure 23: Screenshot of How to Use Page

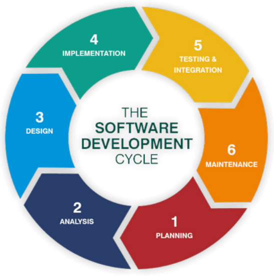


Figure 24: The Software Development Cycle [2]

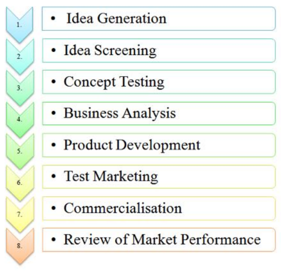


Figure 25: The New Product Development Process [3]

by writing code and testing the module by writing unit tests and conducting end-to-end tests with the platform. In the fourth step, a test is conducted with the client or research participants. In the last step, the feedback from the test is evaluated for the next cycle.

6.3.1 Cycle 1: Assistant Email Address

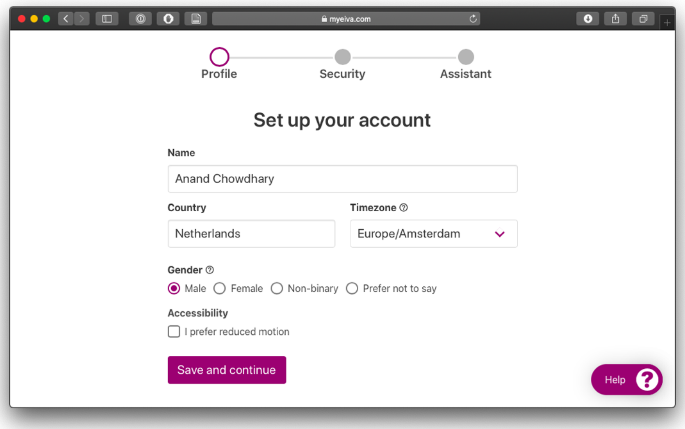
The first cycle was internal with both the initial development and several refactors. One of the specified conditions in Section 3 was to ensure unique email addresses for each organization using the assistant service, because teams may have similar-named assistants. For example, the default name of the assistant, Ara Isaacson (AI), may generate the email address `ara@example.com`, which does not share any information about who the assistant works for. Since unique email addresses for each team is mandatory, the team's slug was incorporated in the address, i.e., `username@example.com`, for example `florian@example.com`.

This has two advantages: firstly, it tells the guests receiving emails from the assistant who the true host is; and secondly, the email receiving architecture can relate the unique email address to the owner's details. However, this might be mistaken as the email address of the user themselves (for example, Florian) rather than their assistant. Therefore, the last change was to add the suffix, `meet-` to each email, e.g., `meet-florian@example.com`.

6.3.2 Cycle 2: Onboarding Interface

In the second cycle, the client's feedback was that users should have a simpler experience when signing up for the EIVA service. Although sensible defaults have been implemented, users may get overwhelmed with several privacy settings, so this should be simplified in an onboarding experience.

Therefore, an onboarding experience was designed where users could easily set their basic information, and select one of three privacy options. Figures 26 and 27 show the updated onboarding interface for user and privacy settings respectively.



Set up your account

Name: Anand Chowdhary

Country: Netherlands

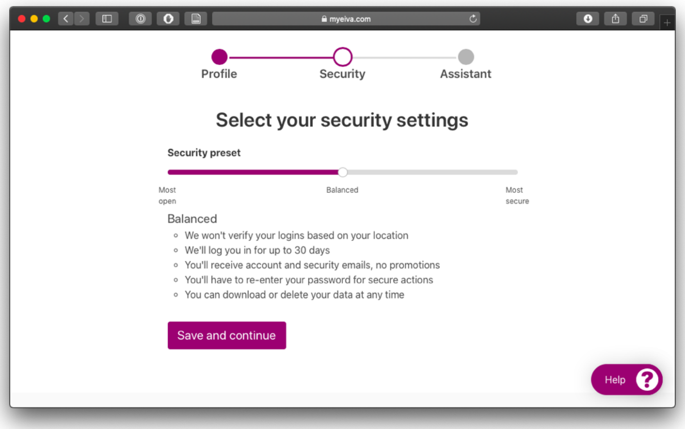
Timezone: Europe/Amsterdam

Gender: [✓ Male] Female Non-binary Prefer not to say

Accessibility: ☐ I prefer reduced motion

Save and continue

Figure 26: Screenshot of Settings Onboarding Page



Select your security settings

Security preset: Balanced

Balanced

- We won't verify your logins based on your location
- We'll log you in for up to 30 days
- You'll receive account and security emails, no promotions
- You'll have to re-enter your password for secure actions
- You can download or delete your data at any time

Save and continue

Figure 27: Screenshot of Security Onboarding Page

A slider interface was implemented for privacy settings, with 'Most open', 'Balanced', and 'Most secure' options. Selecting one of these three options would configure several settings, such as whether to verify logins based on location, whether to receive promotional emails, etc.

6.3.3 Cycle 3: iCalendar Support

The assumption that most users would use Google Calendar ensured that development time was spent on easy connectivity with Google accounts. The official Google Calendar API was used to fetch events when recommending time slots using the open source package `calendar-slots` (Section 6.4). However, this meant that non-Google users could not connect their calendars, and the external client itself uses a self-hosted calendar system. It was therefore recommended to support other calendars that support the iCalendar specification.

Most calendar services are compliant with iCalendar, therefore adding first-class support for iCalendar URLs meant that users using almost any digital calendar could have event conflict resolution features. ICS support was implemented and Google's API was also replaced with the new system. In Figure 22, users are asked to enter an iCalendar URL rather than connecting their Google accounts.

6.3.4 Cycle 4: Dutch Training Data

EIVA only worked in English until Cycle 4, and it was recommended by the client to add natural-language capabilities in multiple languages, especially Dutch. The initial idea was to use a translation API such as Google Translate to first translate incoming emails to English, and then continue processing the email. However, this was troublesome, because imperfect translations did not receive high classification scores in the NLP engine trained with English data.

It was decided to have first-class support for both English and Dutch, and therefore a new Dutch training dataset was added. The process also added a language detection step, so incoming emails in English or Dutch would use their own respective NLP classifiers before continuing with the scheduling process. This ensures equally high accuracy in both languages.

6.3.5 Cycle 5: Bugs and New Domain

In Cycle 5, the client was able to extensively use EIVA and found several bugs and enhancement recommendations. First, it was discovered that deleting the default location (an auto-generated Jitsi Meet link) would throw an error. This was fixed by adding support for custom default meeting locations and automatically changing the default location in case the current default is deleted by the user. Secondly, the client recommended removing HTTP access and redirecting all requests to the secure HTTPS protocol, which was implemented. Several other bugs were also fixed during this feedback cycle.

Also, the website was migrated to the new domain `myeiva.com` (a different domain was used during development), but this had an email deliverability error. Since the domain was purchased on June 1, 2020, email clients were marking emails from EIVA under 'Spam' because the domain was too new. Therefore, the website remained hosted on the new domain, but the assistant emails were moved for the testing period from `@myeiva.com` to `@eiva.o15y.com`. `o15y.com` was purchased over two years ago and therefore had more credibility.

6.3.6 Cycle 6: External Feedback

In the final feedback stage, three experienced web developers — Alex Imbrea, Maurits van der Vijgh , Rohit Bhatia — were asked to use the EIVA webapp and assistant and provide additional feedback. Their suggestions included adding input validation on the client side for the minimum password length of 6 characters, which was only imposed on the server side. They also thought that the intro tour was too long, described as “hand holding”, though they agreed that it was necessary for users without a technical background.

A grammatical error was also found in the outbound email sent by the assistant, and another error that prevented cache invalidation. They were both fixed and deployed before testing with users.

6.4 Open Source Development

Although the preceding project Ara was licensed under the permissive MIT license, EIVA’s source code is made available under the Server Side Public License (SSPL). SSPL makes sure that any further implementations or paid services built around EIVA also have their source code made public, meaning that companies will not be able to sell competing services using EIVA’s publicly-available source code. This was selected to help the future business case.

Nevertheless, several open source projects were built in order to support key EIVA features, and were all licensed under the MIT license. This encourages wide community adoption and contributes back to the open source ecosystem. MIT is the most popular open source license and it allows commercial usage of all code, so competing services are also free to use this package in the future.

Calendar Link For example, the open source project `calendar-link` was built in order to generate event links to popular calendar services like Google Calendar, Microsoft Outlook, and Yahoo! Calendar. This project is now used by many others, receiving over 1,000 downloads every week and contributed to by 5 additional developers. Developers can use the package to programmatically generate links that users can click on to add a specific event to their calendar, with an easy-to-use API.

```
import { google, outlook, ics } from "calendar-link";
const event = {
  title: "My birthday party",
  start: "2020-12-29 18:00:00 +0100",
  duration: [3, "hour"]
};
google(event); // https://calendar.google.com/calendar/...
outlook(event); // https://outlook.live.com/owa/...
ics(event); // BEGIN:VCALENDAR VERSION:2.0...
```

Calendar Slots Similarly, another project `calendar-slots` was released to help find available slots in a user’s calendar. This package can save development times when recommending appointment slots by listing a user’s calendars, fetching all events in a given timeframe, removing slots with conflicts with scheduled events, and recommending a fixed number of slots to the end user:

```
import { findSlots } from "calendar-slots";
const tomorrow = new Date();
const slots = await findSlots({
  slotDuration: 30, // Find 30 minute slots
  slots: 3, // Recommend 3 slots
  from: new Date(), // Starting now
  to: tomorrow.setDate(today.getDate() + 1), // Until tomorrow
});
```

EIVA uses this package under the hood to recommend time slots for appointments. It is also highly configurable, with support for settings such as timezone preference, daily scheduling hours, and custom filter options:

```
import moment from "moment";
const slots = await findSlots({
  slotDuration: 30, // Find 30 minute slots
  slots: 3, // Recommend 3 slots
  from: moment(), // Starting now
  to: moment().add(1, "week"), // Until next week
  days: [1, 2, 3, 4, 5], // Only on week days
  daily: { // Make sure all these conditions are met
    timezone: "Europe/Amsterdam", // Set CET timezone
    from: [9], // Start at 09:00 in the morning
    to: [17, 30], // End at 17:30 in the evening
  }
});
```

Both projects are written in TypeScript and available on the Git hosting service GitHub and JavaScript package registry npm.

6.5 Software Tests

6.5.1 Unit Tests

Unit tests, a method to test individual units of code, are used to ensure all components are behaving as intended [78]. The goal of unit testing is to isolate each part of the code and ensure that individual parts are correct, before writing comprehensive tests for an entire module.

For example, the `dates.spec.ts` file consists of 5 tests for the date parser. Each test provides a strict, written contract that the `findStartEndTime` function must satisfy. This function takes a string of text and returns the start and end time for scheduling. For example, the input "set up an appointment for next week" will return the start time of the following Monday 00:00 and end time of Sunday 23:59, whereas the input of "set up an appointment for tomorrow" will return the start and end time for the following day.

During development, such methods are used extensively to test the natural language processing by supplying many different strings and programmatically testing their outputs. The testing framework Jest is used with an npm script for automated testing.

6.5.2 Continuous Integration

A Continuous Integration (CI) process consisting of unit tests and a build test has been set up to ensure that no bad code gets deployed to users [79]. Whenever a new commit is pushed to the central repository on GitHub, the code is only deployed to the production app if there is a successful build [80].

GitHub Actions is used as the API for CI, and is free for public repositories. A workflow consisting of 5 steps is triggered on every push, using Ubuntu 18.04 LTS, a popular Linux distribution:

1. Checkout the repository (using `actions/checkout@v2`)
2. Setup Node.js v14.3 (using `actions/setup-node@v1`)
3. Install dependencies (`npm ci`)
4. Run all unit tests (`npm run test`)
5. Compile the TypeScript project (`npm run build`)

When there is an error in the build such as if a unit test fails or there is a TypeScript compilation error, the pipeline fails and developers get a notification for error mitigation. On average this CI process takes 2 minutes to complete, and Figure 28 shows the duration of each workflow run.

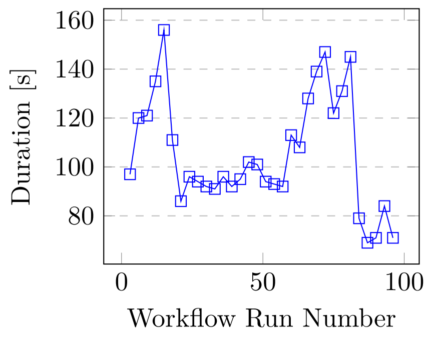


Figure 28: Running Time of Continuous Integration Workflows

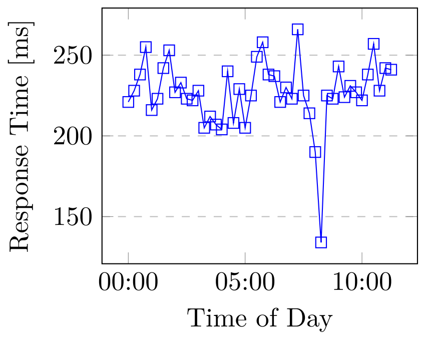
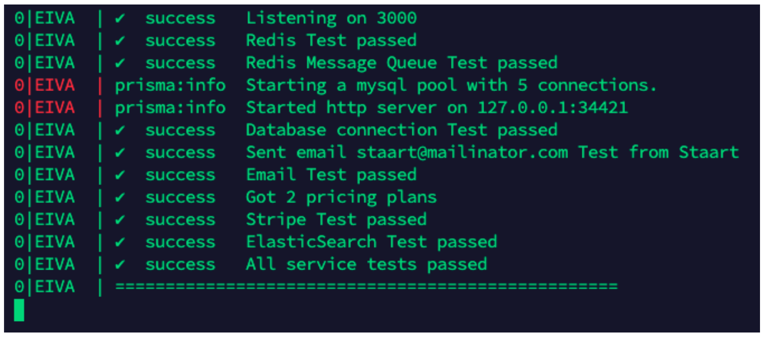


Figure 29: 24-Hour Response Time of the EIVA website

6.5.3 Initialization Tests

Lastly, initialization tests are used after deploying the app to ensure that all services are functioning. The `init-test.ts` file has 6 tests that run as soon as the app is launched. If there is an error with any of these services, it is immediately logged in the console. The output of a successful initialization is shown in Figure 30. After the server is ready, the connection to self-hosted services like Redis and MariaDB are tested; then, a test email is sent to ensure outbound emails are delivered; finally, the connection to third-party services like Stripe and ElasticSearch is tested.



```
0|EIVA  | ✔  success   Listening on 3000
0|EIVA  | ✔  success   Redis Test passed
0|EIVA  | ✔  success   Redis Message Queue Test passed
0|EIVA  | prisma:info  Starting a mysql pool with 5 connections.
0|EIVA  | prisma:info  Started http server on 127.0.0.1:34421
0|EIVA  | ✔  success   Database connection Test passed
0|EIVA  | ✔  success   Sent email staart@mailinator.com Test from Staart
0|EIVA  | ✔  success   Email Test passed
0|EIVA  | ✔  success   Got 2 pricing plans
0|EIVA  | ✔  success   Stripe Test passed
0|EIVA  | ✔  success   ElasticSearch Test passed
0|EIVA  | ✔  success   All service tests passed
0|EIVA  | ==================================================
```

Figure 30: Successful Initialization Tests

6.6 Deployment

Amazon Web Services (AWS) resources are used for deploying the backend project. AWS is highly scalable and has an easy-to-use admin portal to manage resources and billing [81]. AWS is also one of the most secure cloud computing platforms [82]. For minimal latency and data storage compliance, the Frankfurt AWS Infrastructure Region was selected. Therefore, no data ever leaves the European Union.

6.6.1 Infrastructure Cost

The total infrastructure cost between May and July 2020 was approximately US$300. However, the entire cost was subsidized in the form of AWS credits, generously provided by Oswald Labs, the researcher's company, with the exception of the domain name [83]. The `myeiva.com` domain was purchased through Hover and costed US$12.99. All AWS services were billed from May 21, 2020 to July 21, 2020.

Server A dedicated server is used for deploying the EIVA backend. This uses AWS Lightsail which provides an easy-to-use interface for AWS EC2, with the following instance details. The cost of this server was US$20 per month:

- RAM: 4 GB
- Processor: 2 vCPUs
- Storage: 80 GB SSD

ElasticSearch A dedicated ElasticSearch instance is used to store server logs and track usage events, with the following instance details. The total cost of this instance was US$272, billed as US$0.186 per hour:

- Elasticsearch version: 7.4
- Instance type: r5.large.elasticsearch
- EBS volume size: 10 GiB

6.6.2 Backend APIs

To deploy the backend APIs, a custom shell script was written that fetches code from the git repository, installs dependencies, and runs the project. The `update.sh` file consists of the following lines:

```
git pull --force # Pull code from GitHub
npm install # Install npm dependencies
npx prisma generate # Generate database schema
npm run build # Compile TypeScript project
pm2 flush # Clear current deployment's logs
pm2 restart all # Restart all services
pm2 logs # Show the logs of this deployment
```

6.6.3 Frontend Web App

Deploying the frontend web app is much easier because of the CI process. The frontend is hosted on the static site hosting platform Netlify, which automatically builds and deploys the app on push. This service is also free.

6.6.4 Server Monitoring

Uptime Monitoring Although measures have been taken to make sure EIVA services don't go down, such as installing Cloudflare' Always Online which serves a static version of a webpage if it is not available from the origin server [84], there is sometimes nothing one can do to prevent a website from going down. "Traffic spikes, malware attacks, data center problems, etc." can cause server downtime [85].

To monitor the uptime of the services and get real-time notifications in case of downtime, Uptime Robot is installed [86]. In intervals of 5 minutes, Uptime Robot makes network requests to the frontend and backend URLs, logs the response time (Figure 29), and sends an email and Slack notification in case any URL is unavailable.

Over the course of testing weeks, no downtime was recorded for the backend, but the frontend was unavailable twice for a period of 2 minutes and 57 seconds and 2 minutes and 44 seconds on June 7 and June 9 2020 respectively. This was due to a Netlify deploying error because of renamed repositories. Both times, this error was immediately fixed because of the real-time notifications feature, and resulted in a $> 99.98\%$ overall uptime.

Resource Monitoring To make sure the service is not overused and to prevent crashes, Netdata is used to monitor all aspects of the server. Netdata can visualize and monitor real-time metrics, optimized to accumulate all types of data, such as CPU usage, RAM usage, and disk activity [87]. NetData is also open source and consists of a daemon that collects data and an interface to visualize that data [88].

Error Tracking To catch errors in production, Sentry is deployed in both the frontend and backend applications. Sentry is an open source error tracking system [89] that automatically reports errors as soon as they occur. "In addition to sending developers alerts, Sentry also gives them context for potential root cause analysis to help them identify the source of the error" [90].

Although the backend project already collects server logs, they are frequently intentional debugging data. Logging provides a trail of events, but Sentry only focuses on exceptions, i.e., catching application crashes or HTTP request errors. Furthermore, Sentry helps list a trail of usage events like mouse clicks and network requests to discover the context of an error [91].

Licensed under Apache 2, Sentry is open source but offers a hosted service with a generous free plan, which is used for this project. The @sentry/browser package is used to track errors originating in client-side web applications running in browsers, and @sentry/node is used for the Node.js package.

Over the course of the month of June 2020, 3 errors were tracked for the frontend project, but no errors were found in the backend project.

7 Evaluation

The goal of this user experience evaluation was to allow 30 users to create their own assistants using the EIVA service and use the companion app, interact with their assistants by scheduling an appointment, and then hearing their feedback. The form opened on June 1, 2020, and was closed on June 24, 2020, when 30 responses had been recorded.

7.1 Introductory Survey

A diverse group of users participated in the survey, both working professionals and students. 18 out of 30 respondents were professionals employed full-time by a company or organization, another 9 were students, and 3 out of 30 were freelancers or self-employed individuals (Table 6). This healthy diversity of professions helped understand how different categories of users use EIVA.

Table 6: Professions of Participants

Profession	Count
Employed	18
Student	9
Self-employed	3

Table 7: Preferred Modes of Scheduling

Mode	Count
Send an email	24
Work chat message	12
Go to office or desk	12
Phone call	9
Consumer chat	8

Users were also asked their preferred modes of scheduling. The results show a preference for emails, as noted in Section 2. In Table 7, 'Work chat' refers to instant messaging services like Slack and Microsoft Teams whereas 'Consumer chat' refers to services like WhatsApp and Facebook Messenger, and participants could choose multiple options. Apart from email, work chat services or physically reaching out were popular as well, whereas phone calls and consumer chat services are used least frequently. It is clear that email is the preferred form-factor for scheduling appointments, although services like Slack that aim to replace traditional email are also catching up quickly.

7.1.1 Email and Calendar Usage

To further validate that email is the ideal use case for scheduling, participants were asked about their frequency of sending emails (Table 8). More than half of all participants send multiple emails per day, whereas almost one-third send an email once every few days. Only 3 and 2 respondents send an email once every few weeks or per month respectively. Similarly, they were asked about their frequency of checking their email account (Table 9). Almost two-thirds of all participants check their emails multiple times per day, and 7 out of 30 said that they check it at least once per day. 4 and 1 participant(s) check their email at least weekly and monthly respectively. Therefore, a large majority of all participants both send and check emails often.

Table 8: Frequency of Emails Sent

Number of Emails	**Count**
One or more per day	16
One every few days	9
One every few weeks	3
One every month	2

Table 9: Frequency of Checking Email

Number of Times	**Count**
Multiple times per day	18
At least once per day	7
At least once per week	4
At least once per month	1

Since EIVA supports both English and Dutch, participants were asked their preferred languages for email communication. All 30 participants selected English, 9 participants also chose Dutch, and 1 participant also chose German. 1 participant manually entered Polish as well. This suggests that building an English-only service would work for most use cases, but since almost one-third of all users also want to use Dutch, it should be treated as a first-class language. In the future, support for other popular languages can be added as well.

For better integration with large email providers, participants were also asked about their primary email service provider. Over half of all participants use Google's Gmail or G Suite (17), 8 use a Microsoft service (like Outlook, Hotmail, or Live), 2 use a custom-hosted SMTP service, 1 uses Yahoo! Mail, and 2 respondents selected 'Other'. This is consistent with other research, for example Gmail and Outlook were placed first and second in Statista's list of most-popular email services in 2019 respectively.

Furthermore, participants were asked whether they have email notifications enabled on their smartphones, to which all but three participants responded in the affirmative. This shows that an overwhelming majority of users not only check their emails frequently, but also receive real-time email notifications.

Table 10: Email Languages

Language	**Count**
English	30
Dutch	9
German	1
Polish	1

Table 11: Email Service Providers

Service	**Count**
Google	17
Microsoft	8
SMTP	2
Other	2
Yahoo!	1

Apart from scheduling, there are also several other reasons people use email. Participants were asked *What do you use email for?*, and 29 out of 30 participants selected work-related communication. 16 participants also chose scheduling appointments, while 5 use email to connect with friends and family. 3 participants use email for outbound marketing, whereas 1 also uses it for customer support. Therefore, scheduling appointments is only one of the many use cases for email, and the feature set of EIVA can be increased to support them. For example, automated outbound marketing or followups can be implemented.

Although EIVA connects with any calendar application that follows the iCalendar specification (which all major services do), it also directly integrates with

Table 12: Reasons for Sending Emails

Reason	**Count**
Other work-related communication	29
Scheduling appointments	16
Connecting with friends and family	5
Sending marketing emails or newsletters	3
Contacting customer support	1

Google Calendar using the official API. iCalendar support was added in Cycle 3, Section 6.3.3, until when only Google Calendar was supported. This was chosen with the hypothesis that it is the most popular calendar service, which was proved to be true based on the participants' responses. Almost two-thirds of all participants (19) selected Google Calendar as their primary calendar service, 9 out of 30 participants use Microsoft Outlook, and another 8 use Apple Calendar. 3 participants used other calendar services. Lastly, 1 respondent each uses a self-hosted calendar or no calendar.

Table 13: Primary Calendar Services

Calendar Service	**Count**
Google Calendar	19
Microsoft Outlook	9
Apple Calendar	8
Other calendar app	3
Self-hosted calendar	1
I don't use a calendar app	1

7.1.2 Personal Assistant for Scheduling

In the second part of the Introductory Survey, participants were asked hypotheticals about using an assistant for scheduling meetings. The results of the six polar questions (yes/no) are tabulated in Table 14. When asked whether they would ask their human assistants to schedule appointments, all but three responded in the affirmative; and when asked whether they'd ask their AI-powered virtual assistant to do the same, all but six said 'Yes'. The most splitting question was whether EIVA should inform recipients that it is not a human assistant, to which 19 users said 'Yes' and 11 users said 'No'.

Currently, EIVA only sets appointments but the big picture goal of EIVA is to automate several other workflows, like automatically responding to emails or sending outbound marketing emails. Participants were also asked what other features they would like to see. Table 15 summarizes the most-requested additional features that participants want EIVA to have. 22 out of 30 respondents requested reminders, 17 out of 30 selected email follow-ups to team members, 15 out of 30 selected automatic frequent email responses, and 6 participants re-

Table 14: Responses to Polar Questions

Question	Yes	No
If you had a personal assistant, would you ask them to schedule your appointments for you?	27	3
Would you use the EIVA service?	24	6
Should EIVA inform recipients that it is not human?	23	7
Should recipients be able to unsubscribe from EIVA emails?	19	11
Would you like to know whether someone has seen an email sent by EIVA?	22	8
If you have enabled email tracking, should recipients know that a read receipt has been shared?	21	9

quested sending marketing emails. Two participants also added custom requests "Send me my schedule at the start of my day" and "Sort relevant emails, make summaries of long emails, and create a quick overview to-do list".

Table 15: Most Requested Additional Features

Feature	Count
Reminders	23
Send email follow-ups	17
Auto-respond to frequent emails	15
Send marketing emails	6

7.2 Product Usage

Data stored in ElasticSearch between June 1 and June 24 is analyzed, in which 30 participants used the product and responded to the questionnaire. In this period, a total of 2,244 pageviews and 19,177 API logs were collected.

7.2.1 Web App Traffic Analysis

A JavaScript snippet was used to send information about each pageview to the ElasticSearch engine via the API, and most data was computed on the client, such as by parsing the user agent. Data from the 2,244 pageviews is analyzed in this section.

Location Users from four continents used the web application — Asia, Europe, Africa, and North America. The approximate location of users is determined based on their IP address using Maxmind's GeoIP2. Figure 31 and Table 16 highlight the twelve countries users visited from, with the highest participation from India and the Netherlands, followed by Kenya, Poland, the United States, and the United Kingdom. This shows that a truly global participation was observed during this research. A small number of pageviews (<20) were also recorded from Iran, Indonesia, Germany, Finland, France, and Russia.

Furthermore, in the Netherlands, usage was observed in 7 out of 12 provinces, with the maximum usage coming from Flevoland, followed in order by Overijssel, Gelderland, Utrecht, South Holland, North Holland, and Friesland.

This was unexpected because both the researcher and the client are located in Overijssel, but participation from Almere Stad was the highest in the Netherlands, responsible for all of Flevoland's pageviews. Almere Stad was followed by other popular cities — Enschede (99), Deventer (48), Nijmegen (39), and Woudenberg (33). Interestingly, Almere Stad was the second most-popular city during the research, after Gurgaon (418) and before Noida (226), both in the National Capital Region of Delhi, India. Table 17 and Figure 33 show the usage distribution in Dutch provinces, and Figure 32 shows an overview of usage coming from the continent of Europe.

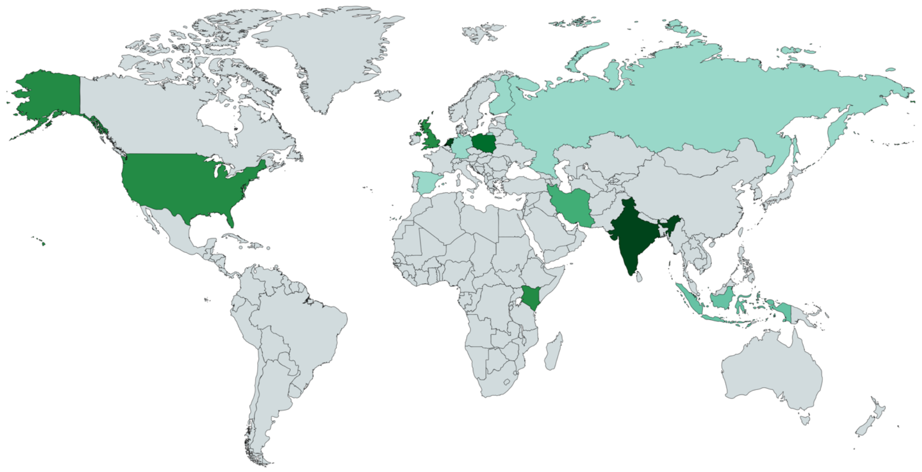


Figure 31: Usage by Country (Worldwide)

Browsers and Devices The User-Agent request header is a characteristic string that provides insight into user' browsers and devices, depending on their privacy settings. In Tables 18 and 19, usage by device type and web browser is displayed respectively, with the exception of requests due to uptime monitoring (Section 6.6.4). Although Google Chrome is the most popular web browser as reported by Wikimedia in November 2019 [92], it was placed behind Mozilla Firefox in usage by participants in this research. After Chrome followed Safari, Safari for mobile, and others. 'Chrome-related' is a sum of pageviews from headless Chrome, Chrome webviews, and the open source Chromium.

Other Analytics In terms of the user's browser language, English was the overwhelming majority, detected in all but 35 pageviews. This was divided into

Table 16: Usage by Country

Country	Pageviews
India	986
The Netherlands	666
Poland	237
Kenya	118
United States	109
United Kingdom	92
Iran	18
Indonesia	6
Germany	4
Finland	4
France	2
Russia	2

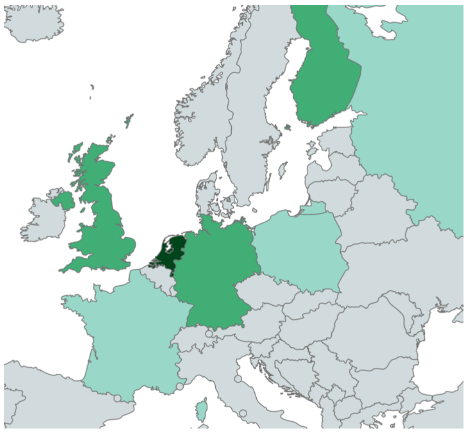


Figure 32: Usage by Country (Europe)

Table 17: Usage by Dutch Province

Province	Pageviews
Flevoland	363
Overijssel	183
Gelderland	39
Utrecht	33
South Holland	18
North Holland	3
Friesland	1

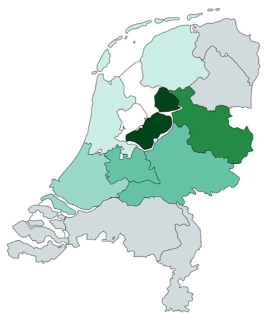


Figure 33: Usage by Dutch Province

Table 18: Usage by Operating System

Operating System	Pageviews
macOS	992
Other Linux	529
Windows	377
Android	217
iOS	83
Ubuntu	44

Table 19: Usage by Web Browser

Browser	Pageviews
Firefox	772
Chrome	726
Safari	637
Safari (mobile)	59
Chrome-related	34
Edge	3

1,311 pageviews in US pageviews, 685 in general English, 206 in British English, and 7 in Indian English. Apart from English, 25 were in Dutch, 8 in Polish, and 2 in Russian. This is positive validation to keep the UI in English.

Similarly, over 99% of all pageviews were done on URLs on the secure HTTPS protocol, with only 4 requests using HTTP. This was due to the secure protocol redirection described in Section 6.2.5.

Most users used the web app on a large screen, not a smartphone. In fact, over 30 unique screen resolution combinations were tracked. The most common absolute screen resolution was the high definition 1920x1080 (35%), followed by 1440x900 at almost a quarter of all pageviews. Interestingly, 2% of all pageviews still used the outdated 800x600 resolution.

7.2.2 API usage

Whenever a participant uses the web app, requests to the API are made. An average of 799 API requests were made on each day. The number of requests on each day is graphed in Figure 34. An API endpoint is triggered when a user sends a new email to their assistant, as described in Section 6.1.9. Unless an error occurs, the assistant responds to each email as well.

Figure 35 shows the number of new emails processed every day. The average number of daily emails processed per day was 7.5 with a total of 98 emails. It is visible that participation was greatest in the beginning, gradually slowed down, and then increased again at the end. This is because invitations to participate were sent both in the beginning (June 1) and later in the research (June 21).

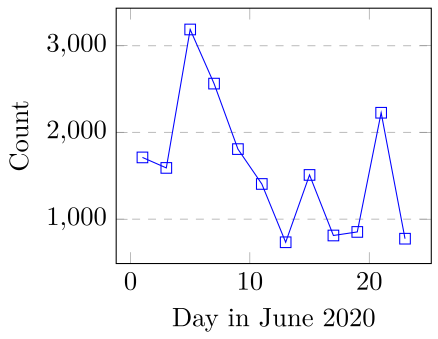


Figure 34: Number of API requests received per day

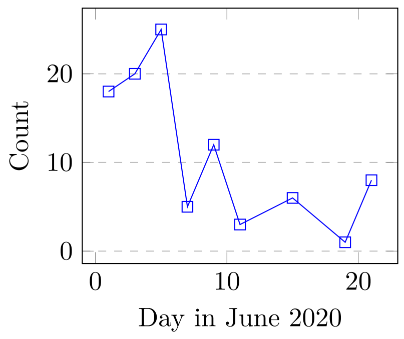


Figure 35: Number of emails processed per day

7.2.3 Heatmaps

To better measure how participants used the web app, each mouse click was also tracked. This tracking was completely anonymous, collecting only the details of the component clicked on apart from the datapoints described above, without linking the clicks to a particular user ID. Therefore, this approach is both privacy-conscious and useful to understand whether users interact with the interface in the intended way.

Heatmaps convey the distribution of attention on a page [93]. A total of 4,940 clicks were tracked and the Figures 36 and 37 are presented by superimposing the heatmap over the screenshot of the respective component. A greater green color on a point refers to an increasing number of clicks.

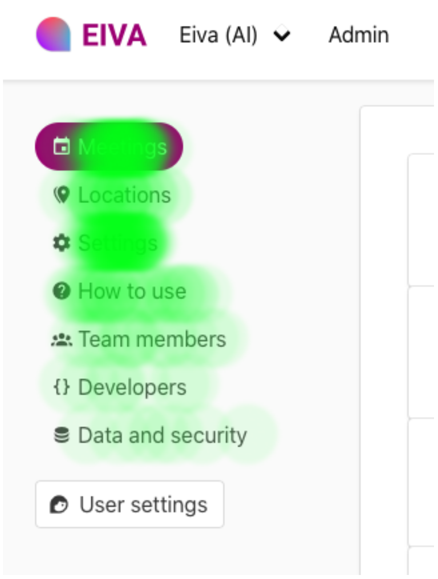


Figure 36: Heatmap for Sidebar

Navigation In Figure 36, the distribution of clicks on the sidebar navigation is showcased. This heatmap was generated using 403 datapoints and shows that Settings was clicked on most frequently (106 times), followed in order by Meetings (105), Locations (71), How to use (58), Team members (13), Developers (20), and Data and security (15). This is consistent with the introductory guide, that led users to the first four links. It also shows that some users chose to go to other pages that were not part of the tour as well.

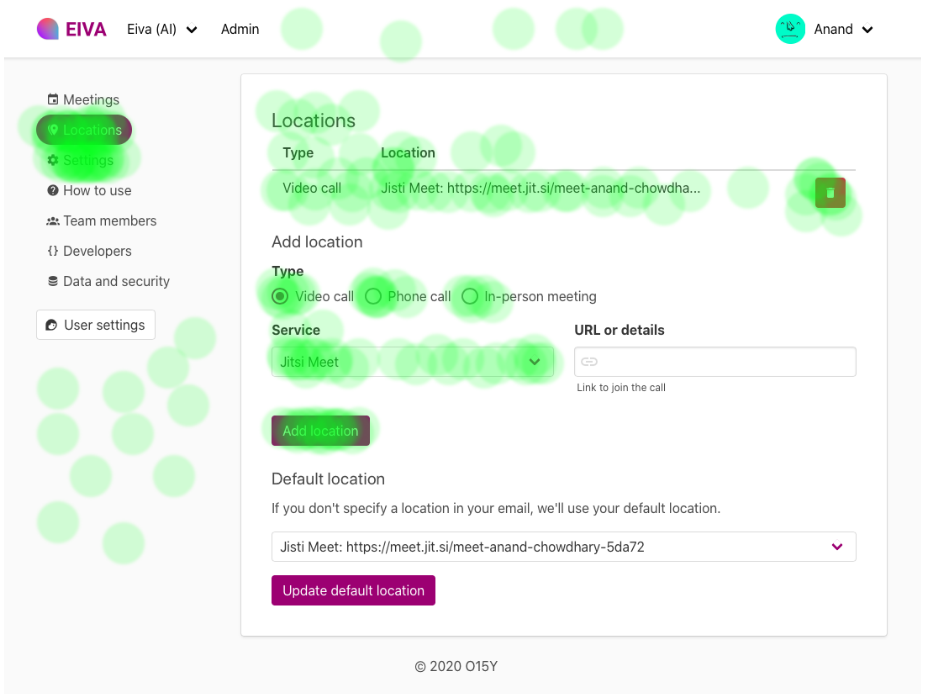
Webpage The Locations page is an ideal example to measure clicks because it has several controls and links. Figure 37, generated with 237 datapoints, shows that the sidebar navigation on the Locations page has the most frequently clicked item of Settings (19 clicks), which is expected since it is the next link after Locations itself. Users also clicked on the table containing the list of saved locations and the Delete button for each item.

In the Add Locations selector, both Video call and Phone call were selected 6 times, whereas In-person meeting was selected 5 times. The Service dropdown selector was clicked on 24 times. The Add Location button was clicked on 14 times. In this example, the Update default location button is unused because it was added after June 10, 2020. Users also clicked in the sidebar below the navigation a few times. Overall, it is clear that users interacted with all controls on the webpage — the sidebar navigation, locations table, buttons, and input fields. The ‘Default location’ section was added after testing, so it has no clicks, as expected.

7.3 User Experience Evaluation

The first question, *What comes to mind when you think about EIVA; how would you describe it to a friend?*, allows users to share their first impressions after using the service. Some of the responses were:

1. “A smart interface to my calendar”
2. “A way to avoid having to talk to people to schedule your appointments with them”



Locations

Type	Location
Video call	Jisti Meet: https://meet.jit.si/meet-anand-chowdha...

Add location

Type

[✓ Video call] Phone call In-person meeting

Service: Jitsi Meet

URL or details:

Link to join the call

Default location

If you don't specify a location in your email, we'll use your default location.

Jisti Meet: https://meet.jit.si/meet-anand-chowdhary-5da72

© 2020 O15Y

Figure 37: Heatmap for Locations Page

3. "Make my appointment very easy, Easy to see all meetings in one screen! Awesome Product!"
4. "Interesting personal automation, severely limited at the moment"

7.3.1 Webapp Experience

Users were asked to rate the web app on different parameters, from 0 to 5. All parameters scored over 4 out of 5, with privacy features rated with the highest average of 4.7, and the introductory tour rated the lowest average of 4.2. The interface design was rated 4.6, its functionality was rated 4.5, and its ease of use was rated 4.4. Apart from the average, the median, mode, and standard deviation of each rating was also computed. The results overwhelmingly suggest that users enjoyed web app.

7.3.2 Assistant Experience

Similarly, users were asked to rate the assistant on different parameters, from 0 to 5. All but two parameters scored an average greater than 4, with the lowest-rated question being *How much do you trust the assistant to not make mistakes?*. Participants rated the overall experience as 4.5 out of 5, and customizability and flexibility as 4.4 each.

Table 20: Webapp Experience Ratings

Parameter	**Average**	**Median**	**Mode**	**Std. Dev.**
Overall experience	★★★★⯪ 4.4	5	5	0.7
Design	★★★★★ 4.6	5	5	0.7
Functionality	★★★★⯪ 4.5	5	5	0.6
Ease of use	★★★★⯪ 4.4	4.5	5	0.7
Privacy features	★★★★★ 4.7	5	5	0.5
Onboarding flow	★★★★⯪ 4.5	5	5	0.7
Introductory tour	★★★★☆ 4.2	4	5	1.0

Table 21: Assistant Experience Ratings

Parameter	**Average**	**Median**	**Mode**	**Std. Dev.**
Overall experience	★★★★⯪ 4.5	5	5	0.9
Customizability	★★★★⯪ 4.4	5	5	0.8
Flexibility in understanding language	★★★★⯪ 4.4	5	5	0.9
Sounding human-like over email	★★★★☆ 4.1	4	5	0.9
Trust to not make mistakes	★★★⯪☆ 3.6	4	4	1.2
Time recommendation accuracy	★★★★☆ 3.9	4	4	1.0

To participate in the research, users were required to send at least one email to EIVA for scheduling an appointment. They were free to send multiple emails, and were asked about the number of emails they sent. 10 out of 30 participants sent between 2 and 5 emails, while the other 20 only sent the one.

To find out whether the assistant understood the overall meaning of their email in natural language, participants were asked *Did the assistant understand your email(s) correctly?*. 24 out of 30 participants responded in the affirmative and 5 out of 10 responded with 'Partially', while only one participant responded in the negative. This suggests that EIVA did a well-enough job understanding incoming emails most of the time.

Table 22: Number of Emails Sent

Number	**Count**
1	20
Between 2 and 5	10
More than 5	0

Table 23: Correctly Understood

Response	**Count**
Yes	24
Partially	5
No	1

Apart from understanding the meaning of an email, participants were asked whether EIVA recommended the correct location for the meeting. The location recommendation was correct for 24 out of 30 participants and incorrect for 5 participants. 1 participant chose to not answer this question. Because of the API webhook, EIVA responds to emails almost instantly. When asked about whether the assistant responded to emails in a reasonable amount of time, all 30 participants responded with 'Yes'.

They were also directly asked whether EIVA met their expectations in terms of scheduling an appointment. All but one participants responded positively.

Table 24: Location Recommendation

Accurate	**Count**
Correct	24
Incorrect	5

Table 25: EIVA Met Expectations

Response	**Count**
Yes	29
No	1

To test the hypothesis that EIVA is indistinguishable from a human assistant, users were asked *If you were the recipient of an email from EIVA, would you be able to tell whether it's an email from an AI or human assistant, if you didn't already know?*. 11 out of 30 participants said that they would be able to tell, while 18 out of 30 said that they wouldn't. 1 participant chose not to answer. This shows that for most users, EIVA would essentially act as a human assistant.

7.3.3 Business Case

Since the big picture goal of the client is to launch EIVA as a paid service, participants were asked whether they would use EIVA on launch. 25 out of 30 participants said that they would, whereas 5 said that they wouldn't. This shows a high user retention rate after usage, with 83% of all users wanting to continue to use the assistant service.

Participants were also asked how much they would be willing to pay every month for such a service. 6 out of 30 participants said that the service should be free, 13 out of 30 would be willing to pay up to €5 per month, while 10 out of 30 would pay up to €10 per month. One participant wanted to pay €20 or more per month. Although it seems like the average amount users would want to pay is up to €6.16 per month, this includes all professional categories of respondents. Calculating individually, the average for students is up to €4.44 per month, for employees of a company or organization, it is up to €6.38 per month; and for self-employed professionals or freelancers, it is as high as €10 per month.

Table 26: Would Use EIVA

Accurate	**Count**
Yes	25
No	5

Table 27: Monthly Subscription Fees

Response	**Count**
€0	6
€5 or less	13
€10 or less	10
€20 or less	0
More than €20	1

7.3.4 Final Impressions

Respondents could optionally add additional information in the form of long-form answers. A selection of answers with analysis is listed below, and all answers are available in Appendix 3.

When asked *What do you like the most about EIVA?*, 5 users mentioned the ease of use and 3 users mentioned the fast response time. Users also liked the overall concept, security features, and the fact that EIVA doesn't interfere with their normal workflow. They also appreciated the fact that they don't have to manually select time slots. Users also liked the natural-language conversation and the user interface. Some quotes from participants included "[EIVA is] easy to handle and helps you to save your time and energy," and "[EIVA] takes the most painstaking part of the workhours off [your] hands".

On the contrary, responses to the question *What do you like the least about EIVA?* included "pretending to be human", the "lengthy and slightly technical" onboarding process, and the mobile interface. One user also said that "[EIVA] doesn't fully seem to know/understand my reasoning behind picking or proposing certain meeting times, where maintaining focus during my work day plays an important role." This suggests that more fine-grained control over recommended time slots should be allowed in the web app interface.

Tables 28 and 29 analyze responses to questions *Did you find anything frustrating that you wish was easier or different?* and *Is there anything that you wish the assistant could do, in terms of scheduling, that it doesn't currently?* respectively.

Lastly, users had the option of adding any additional feedback, suggestions, or comments, which almost half of all respondents chose to add. One user said "I really like EIVA, and I think that we need such kind of smart applications in today's world that do not only save time, but also help us act more efficiently in other work as well." Users also stated that EIVA is "great and extremely easy to use" and has "great potential".

Table 28: User Frustrations

Response	Analysis
Dutch grammar can be improved	As a non-native speaker, Google Translate was used for Dutch text in emails
Onboarding is too long or should be skippable	The current onboarding flow takes users through each setting and page; this was to ensure users know of all available options, but can be understandably frustrating
Time recommendations should take email into account	EIVA initially recommended there time slots in the following week, but has since improved to understand email content and recommend accordingly
Onboarding starts again on second login	The onboarding completion status is stored in local storage, so logging in from another device will ask users to complete the tour again
Adding calendar connection URL is hard	Users have to know how to generate an iCalendar URL from their calendar client of choice, so documentation should be added to make that process simpler
Slow activation email	The cron to send emails is scheduled for every minute, so it can take up to 60 seconds to send an email to users

Table 29: Feature Recommendations

Recommendation	Analysis
Create a meeting without having to send an email	The web app can include an option to manually schedule meetings
Support SMS or WhatsApp instead of email	Currently, EIVA only works over email, and more form factors can be added over time
Send daily schedule summary emails	This was a feature part of Ideation, but was not implemented due to time constraints and should be added
Invite multiple guests in each appointment	Users can already invite multiple guests, but slot selection UI has to be improved
Select preferred meeting time slots manually	This can be added by providing a preference UI for each slot
Increase number of recommended daily slots	A customizable setting can be added with the number of recommendations to send
Cancel or reschedule meetings from the web app	This was another feature part of Ideation that wasn't implemented and should be added

8 Conclusion

A functional Email-based Intelligent Virtual Assistant (EIVA) was developed along with its companion web app using TypeScript with Node.js and Vue.js, and with a focus on industry best-practices and customizability.

30 users from around the world participated in this research, with a majority coming from India and the Netherlands. Most participants were employed full-time, many were students, and some were freelancers. This healthy diversity of professions helped understand how different categories of users would use EIVA. The hypothesis that email is the most common mode for scheduling appointments was supported by the results of the Introductory Survey. Participants used a variety of devices, operating systems, and web browsers to use the web app, and sent almost 100 total emails to their assistants. 19,177 API requests from 2,244 pageviews were tracked, and heatmaps for mouse clicks were also generated to measure the distribution of attention on webpages.

Participants responded very positively after using the assistant and the web app, with overall experience ratings of 4.5 and 4.4 out of 5 respectively. The privacy features were most highly rated, and users also enjoyed the design of the interface, the onboarding flow, and the ease-of-use. The lowest ratings were for time recommendation accuracy and trust in EIVA not to make mistakes. This suggests that users are excited about the product, but there is still work to be done to increase the assistant's utility and accuracy. Respondents also shared their frustrations and recommendations, which were analyzed for future work. All but one participants said that EIVA met their expectations, and 25 out of 30 would use it in the future if it launches as a service. Most would also be willing to pay for it, with an average amount up to €6.16 per month.

Since most users are already used to using voice assistants in their day-to-day lives, they were open to adopting a virtual assistant for scheduling as well. Although built with a basic natural language processing system, EIVA was accurate- and intelligent-enough to recommend the correct parameters in the vast majority of cases. Based on the positive response, development work will continue and EIVA will launch as a service for professionals later this year.

9 Future Work

9.1 Features and Improvements

In Section 7.3.4, participants shared their frustrations which included enhancements and bugs, which should be fixed. For example, Dutch grammar should be improved with the help of a native speaker, onboarding time should be reduced, documentation should be improved, and time recommendations should be smarter and context-aware.

Users also shared feature recommendations which should be implemented in the future. New features such as setting up appointments without sending emails and support for other forms of communication like instant messages and SMS should also be added. Users are also currently unable to manually select recommended slots, or cancel or reschedule meetings. Furthermore, support for multiple guests also needs several improvements.

In terms of technical updates, a bloom filter may be used when searching for invalidated tokens, which will make the search faster and more efficient on scale. Prisma, the database toolkit used by the backend, also recently added support for native JSON data types, so the dependencies and source code should be updated with the new syntax. The NLP training data can also be increased to add support for more languages.

9.2 Business Case

In Section 2, it is highlighted that professionals currently either schedule appointments themselves or hire assistants to help with the task. However, regardless of whether the professional or their assistant schedules appointments, the time wasted is not insignificant. According to the European Commission, the average Dutch small or medium-sized enterprise (SME) employs 3.2 people [94]. If each employee attends 7 meetings per week (which is the average for professionals), 26 otherwise productive hours are wasted every month in scheduling meetings [19]. This costs the company over €450 in lost time, based on the average wage of €17.6 per hour [95] This adds up to almost €2.25 billion per year for the SME industry as a whole, and the numbers are even higher for larger enterprises. University of Twente, for example, employs 3,150 professionals, adding up to over €400,000 in lost time per year.

Therefore, a strong business case can be built around launching EIVA as a service. Since the assistant can save around €150 in lost time per month for professionals, a service that costs as high as €100 per month per person has a return on investment (ROI) of 50%. That being said, a pricing point of less than €50 per month can be achieved because of the low cloud infrastructure costs. The proposed pricing plans are:

1. Basic plan for €10 per month, targeted towards students, self-employed young professionals, and early-stage entrepreneurs. This plan will have no support for custom domain names, but will include unlimited scheduling and assistant usage.

2. Pro plan for €25 per month. This will include all features of the Basic plan with support for custom domain names.

3. Team plan for €30 per user per month, targeted towards businesses or institutions who want to onboard their entire team to EIVA for a collaborative experience.

4. Custom plan with custom billing for large enterprises with features like dedicated support, on-premise hosting, and uptime service-level agreements (SLA).

9.3 Impact Assessment

When deploying this service to the market, there can be several ethical and societal challenges that need to be explored, such as AI-driven job loss. Therefore, an impact assessment must be conducted before launch.

Just like in other applications of automation software, loss of employment for assistants is an important social disruption that this product unfortunately encourages. In the interest of saving both time and money, companies may choose to deploy AI-powered assistants on an organization-wide level and terminate the employment of all their secretaries in the future.

In a survey of administrative assistants, all but one reported that most of their work was scheduling-related [15]. Though in the current state, EIVA only focuses on appointment scheduling, this is expected to change in the future as more features are added. Slowly, software agents like EIVA will be able to do more and more of the day-to-day job of an assistant, and will prove to be both faster and more inexpensive.

For the purposes of this thesis project, there is a strong conclusion that as EIVAs become smarter and market adoption increases, there definitely will be a cause-effect relationship with human assistant unemployment. I'd further say that someone who has used EIVA (and perhaps paid around €20 per month for it) will not ever want to go back to a human assistant that is slower, makes more mistakes, and costs over 100x more.

Appendix 1: Information Brochure

If you want to get in touch with the researchers, client, or supervisor, you can find their details below. If you have any queries, complaints, or comments about this research, you can contact the Secretary of the Ethics Committee.

Research Leader Anand Chowdhary
Address: Roerstaart 28, 7543AC Enschede, the Netherlands
Phone: +31 0 644691056
Email: a.chowdhary@student.utwente.nl

Research Client Speakup B.V.
Address: Institutenweg 6, 7521PK Enschede, the Netherlands
Phone: +31 0 887732587
Email: info@speakup.nl

Research Supervisor Dr. Job Zwiers
Address: Zilverling 1060, 7522NH Enschede, the Netherlands
Phone: +31 0 534893816
Email: j.zwiers@utwente.nl

Secretary of the Ethics Committee Dr. Petri de Willigen
Address: Zilverling 1051, 7522NH Enschede, the Netherlands
Phone: +31 0 534892085
Email: ethics-comm-ewi@utwente.nl

Purpose Scheduling appointments manually can waste a lot of time, and our goal is to provide you with a virtual assistant that can automate the process of scheduling based on your preferences and availability. This research will help us understand the accuracy of the assistant, the end user experience, and whether or not end users will use it when we launch it as a service.

Research Procedure After you have signed the Consent Form and answered some questions, you can register for the virtual assistant service on the website. An onboarding interface will help you set up your account, connect your calendar, and set up your scheduling preferences (such as working days, meeting locations, and preferred times). After using the virtual assistant to schedule at least one meeting, you can continue to fill this survey about your experience. Apart from your survey response, computed data such as how long it takes the assistant to set up your appointment and analytics about your usage of the web application will also be collected and aggregated. Once the research is over, you can choose to receive a copy of the results.

10 Minutes It will takes around 10 minutes to complete this survey, including interacting with your virtual assistant. It is recommended to use a laptop, tablet, or desktop computer. If none of these are available to you, you can also use a smartphone. After you have submitted this form, you can continue to use the virtual assistant service, if you choose to, but only the usage within the research period will be measured.

Remuneration This research is voluntary and there is no monetary compensation for participation. However, after this research has ended, this virtual assistant service will launch as a paid subscription service for professionals. As a research participant, you will receive €100 in service credits or equivalent, and a coupon code will be emailed to you at the end of this research.

Anonymous Anonymity will be guaranteed to all participants, and your individual data will not be disclosed to third parties without your explicit permission. Furthermore, the data obtained from this research is not published in any way that would make it possible to link the results or other findings with a particular subject. Only aggregated or anonymized data will be published.

Voluntary Participation remains at all times voluntary and you may refuse to participate in this research at any time, without giving any reason. You can also refuse that we use your data for the research within 24 hours of completing the survey. There is no specific category of persons who are advised not to participate in the research due to an increased level of risk or discomfort.

COVID-19 Note This research is completely online and based on your usage of a web application, sending emails, and answering a survey. At no point will participants be asked to interact with a physical installation or answer questions in person. It is recommended that you participate from the comfort of your home. There is no possibility of accidental discoveries in this research.

Appendix 2: Consent Form

This section ensures that we have your permission to conduct this research. Please note that participation remains at all times voluntary and you may refuse to participate in this research at any time, without giving any reason.

1. Before we sign you up for this research, we need to make sure that you can give us permission. Note that you are not eligible for this research if you are under the age of 18 or otherwise incapable of giving us your informed consent. I am above the age of 18 and capable of giving informed consent.

2. In this research, you will use a virtual assistant web application and communicate with the assistant over email. You will have to answer a series of questions about your experience using the application and communicating with the assistant. By checking "I agree", you permit us to use the data from your answers for purposes of this research.

3. Optional: To find available meeting slots in your agenda, the virtual assistant will require access to your calendar. To allow access, you can add a link to your calendar using the web app. You can also remove access at any time. By checking "I agree", you permit us to have read-only access to your calendar.

4. In order to use the web application, you provide your personal information such as your name and email address. Optionally, you can also add your phone numbers and office addresses to allow the virtual assistant to schedule in-person meetings or phone calls. By checking "I agree", you permit us to store this information and use it for scheduling appointments.

5. Your usage of the web application will be analyzed in the background, such as how long it takes you to complete the onboarding flow and to set up an appointment using the assistant. This data will be only be collected anonymously and published after aggregation. By checking "I agree", you permit this data collection.

6. Under the General Data Protection Regulation, we need your permission to store your information on our servers. You can export a copy of all your data or permanently delete it at any time from the web application. By selecting an option, you permit us to store your information.

Appendix 3: Raw Responses

All raw responses from the survey are available in CSV format on the GitHub project: `https://github.com/AnandChowdhary/bsc-thesis` in `responses.csv`.

The following table shows all collected responses for rating questions. Columns refer to question number while rows are randomized responses. In the last three rows, $\bar{x}$ refers to the mean, $\tilde{x}$ is the median, Mo is the mode, and σ is the standard deviation.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1	5	5	5	5	5	5	5	5	5	5	5	5	5
2	4	4	4	3	5	5	3	4	3	3	4	3	3
3	4	4	4	4	5	4	1	4	4	2	3	4	3
4	5	5	5	4	5	5	3	5	5	5	3	4	4
5	5	5	5	5	5	5	5	5	5	5	3	1	5
6	3	5	4	4	4	4	4	4	4	4	4	4	4
7	5	5	5	5	5	5	5	5	5	5	5	5	5
8	5	5	5	5	5	5	4	5	5	5	5	4	4
9	4	4	4	3	4	3	4	4	5	4	4	3	4
10	3	4	3	3	3	3	3	1	2	2	2	2	2
11	5	5	5	4	5	4	4	5	5	4	4	4	2
12	4	5	4	3	4	4	5	5	4	4	4	3	5
13	4	5	4	4	4	4	4	4	3	3	3	2	2
14	5	5	5	5	5	4	5	5	5	5	5	5	5
15	5	5	5	5	5	5	5	5	5	5	5	5	5
16	5	5	5	5	5	5	5	5	5	5	5	5	5
17	5	5	5	5	5	5	5	5	5	5	5	5	5
18	5	5	5	5	5	5	5	5	5	5	3	1	5
19	5	4	5	5	4	5	5	5	4	5	5	3	3
20	4	5	4	4	4	5	4	5	4	5	4	4	3
21	4	5	4	5	5	5	5	4	5	5	5	3	3
22	5	5	4	5	5	5	5	5	4	5	4	3	3
23	3	2	4	5	5	5	5	5	5	5	5	4	5
24	4	4	4	4	4	3	4	4	3	4	4	3	4
25	5	5	5	4	5	5	3	5	4	4	3	5	4
26	3	4	4	4	4	4	3	4	4	4	4	4	4
27	4	4	5	5	5	5	4	5	5	4	4	3	3
28	4	5	4	4	5	5	4	4	4	4	4	4	3
29	5	5	5	4	5	4	3	5	5	5	5	4	5
30	5	4	5	5	5	5	5	3	5	5	3	2	4
$\bar{x}$	**4.4**	**4.6**	**4.5**	**4.4**	**4.7**	**4.5**	**4.2**	**4.5**	**4.4**	**4.4**	**4.1**	**3.6**	**3.9**
$\tilde{x}$	5	5	5	4.5	5	5	4	5	5	5	4	4	4
Mo	5	5	5	5	5	5	5	5	5	5	5	4	5
σ	0.7	0.7	0.6	0.7	0.5	0.7	1.0	0.9	0.8	0.9	0.9	1.2	1.0

Appendix 4: Code Repositories

All code written for this project is open source, available on GitHub:

1. Backend APIs: `https://github.com/o15y/eiva`
2. Frontend web application: `https://github.com/o15y/myeiva.com`
3. Thesis: `https://github.com/AnandChowdhary/bsc-thesis`
4. Companion open source packages:
   (a) Calendar Slots: `https://github.com/AnandChowdhary/calendar-slots`
   (b) Calendar Link: `https://github.com/AnandChowdhary/calendar-link`
   (c) Staart API: `https://github.com/staart/api`
   (d) Staart UI: `https://github.com/staart/ui`

Appendix 5: Open Source Licenses

This project is standing on the shoulder of giants; it wouldn't be possible without the open source projects listed below. Only direct npm dependencies are listed below (not development dependencies or subdependencies).

Backend

MIT License @staart/config, @staart/disposable-email, @staart/elasticsearch, @staart/errors, @staart/mail, @staart/messages, @staart/mustache-markdown, @staart/payments, @staart/redis, @staart/scripts, @staart/server, @staart/text, @staart/validate, axios, calendar-link, calendar-slots, chrono-node, cron, fs-extra, geolite2-redist, jsonwebtoken, mailparser, maxmind, moment, moment-timezone, mysql, natural, node-email-reply-parser, node-emoji, otplib, qrcode, random-int, systeminformation

Apache 2.0 @prisma/cli, @prisma/client, client-oauth2, googleapis, snyk

Other randomcolor (CC0), email-signature-detector (ISC), @sentry/node (BSD 3 Clause)

Frontend

MIT License @nuxtjs/axios, @nuxtjs/pwa, analytics-icons, countries-and-timezones, js-file-download, jwt-decode, nuxt, nuxt-buefy, shepherd.js, ua-parser-js, unique-selector, vue-stripe-elements-plus, vuex-persist

Other @sentry/browser (BSD 3 Clause)

Appendix 6: Prisma Database Schema

Enumerated types

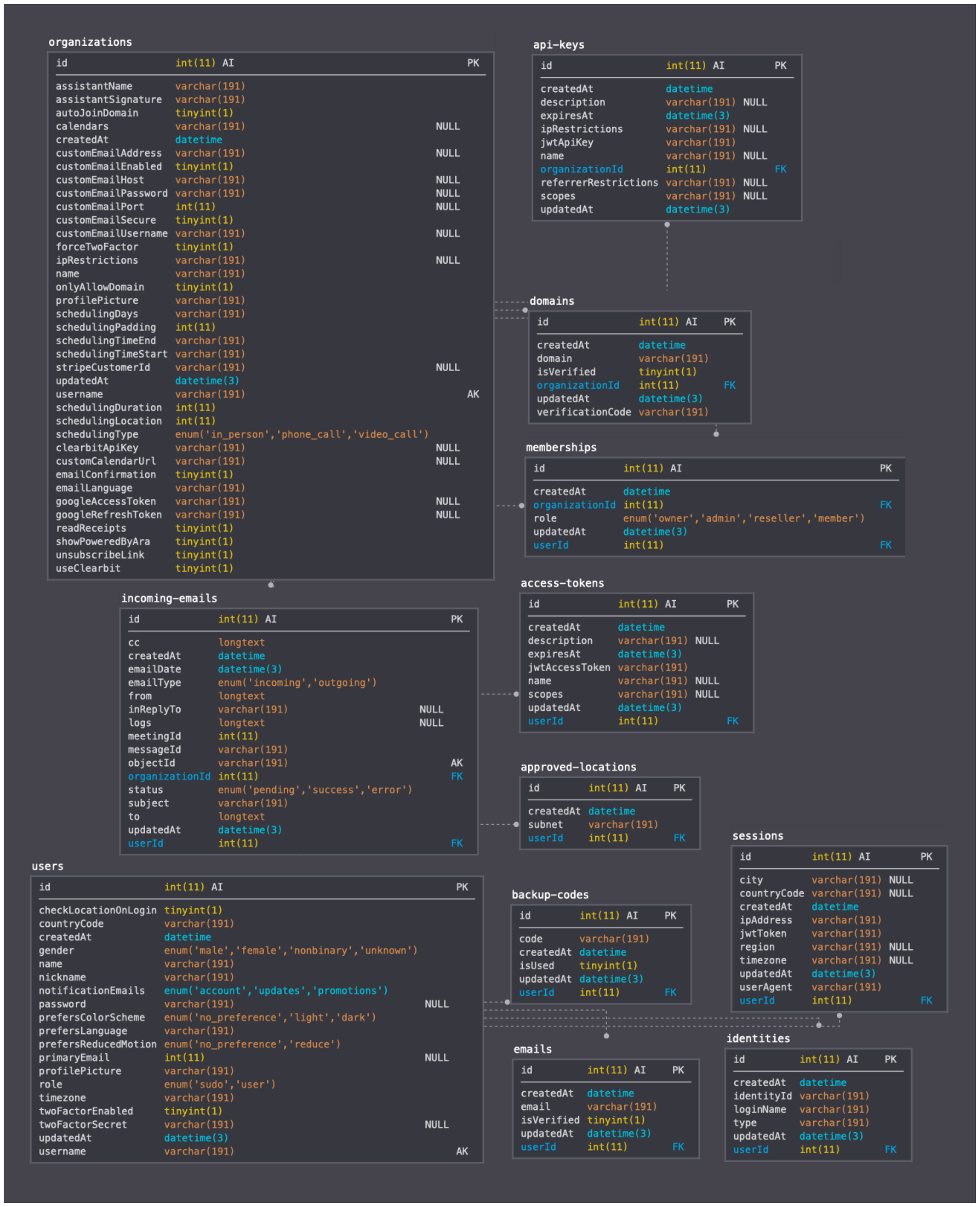
Enumerated types (ENUMs) are a data type consisting of a set of named values. In the Prisma database schema, the following custom-defied ENUMs are used.

```
enum Gender {
  MALE
  FEMALE
  NONBINARY
  UNKNOWN
}

enum NotificationEmails {
  ACCOUNT
  UPDATES
  PROMOTIONS
}

enum PrefersColorScheme {
  NO_PREFERENCE
  LIGHT
  DARK
}

enum PrefersReducedMotion {
  NO_PREFERENCE
  REDUCE
}

enum UserRole {
  SUDO
  USER
}

enum MembershipRole {
  OWNER
  ADMIN
  RESELLER
  MEMBER
}

enum MeetingType {
  IN_PERSON
  PHONE_CALL
  VIDEO_CALL
}

enum EmailStatus {
  PENDING
  SUCCESS
  ERROR
}

enum EmailType {
  INCOMING
  OUTGOING
}
```

Entire Schema

The full schema was omitted from this appendix because it is 357 lines. It is available in the `prisma` directory part of the backend source code, on `https://github.com/o15y/eiva/blob/master/prisma/schema.prisma`.

Appendix 7: Database Relationships



organizations

id	int(11) AI		PK
assistantName	varchar(191)		
assistantSignature	varchar(191)		
autoJoinDomain	tinyint(1)		
calendars	varchar(191)	NULL	
createdAt	datetime		
customEmailAddress	varchar(191)	NULL	
customEmailEnabled	tinyint(1)		
customEmailHost	varchar(191)	NULL	
customEmailPassword	varchar(191)	NULL	
customEmailPort	int(11)	NULL	
customEmailSecure	tinyint(1)		
customEmailUsername	varchar(191)	NULL	
forceTwoFactor	tinyint(1)		
ipRestrictions	varchar(191)	NULL	
name	varchar(191)		
onlyAllowDomain	tinyint(1)		
profilePicture	varchar(191)		
schedulingDays	varchar(191)		
schedulingPadding	int(11)		
schedulingTimeEnd	varchar(191)		
schedulingTimeStart	varchar(191)		
stripeCustomerId	varchar(191)	NULL	
updatedAt	datetime(3)		
username	varchar(191)		AK
schedulingDuration	int(11)		
schedulingLocation	int(11)		
schedulingType	enum('in_person','phone_call','video_call')		
clearbitApiKey	varchar(191)	NULL	
customCalendarUrl	varchar(191)	NULL	
emailConfirmation	tinyint(1)		
emailLanguage	varchar(191)		
googleAccessToken	varchar(191)	NULL	
googleRefreshToken	varchar(191)	NULL	
readReceipts	tinyint(1)		
showPoweredByAra	tinyint(1)		
unsubscribeLink	tinyint(1)		
useClearbit	tinyint(1)		

api-keys

id	int(11) AI		PK
createdAt	datetime		
description	varchar(191)	NULL	
expiresAt	datetime(3)		
ipRestrictions	varchar(191)	NULL	
jwtApiKey	varchar(191)		
name	varchar(191)	NULL	
organizationId	int(11)		FK
referrerRestrictions	varchar(191)	NULL	
scopes	varchar(191)	NULL	
updatedAt	datetime(3)		

domains

id	int(11) AI	PK
createdAt	datetime	
domain	varchar(191)	
isVerified	tinyint(1)	
organizationId	int(11)	FK
updatedAt	datetime(3)	
verificationCode	varchar(191)	

memberships

id	int(11) AI	PK
createdAt	datetime	
organizationId	int(11)	FK
role	enum('owner','admin','reseller','member')	
updatedAt	datetime(3)	
userId	int(11)	FK

incoming-emails

id	int(11) AI		PK
cc	longtext		
createdAt	datetime		
emailDate	datetime(3)		
emailType	enum('incoming','outgoing')		
from	longtext		
inReplyTo	varchar(191)	NULL	
logs	longtext	NULL	
meetingId	int(11)		
messageId	varchar(191)		
objectId	varchar(191)		AK
organizationId	int(11)		FK
status	enum('pending','success','error')		
subject	varchar(191)		
to	longtext		
updatedAt	datetime(3)		
userId	int(11)		FK

access-tokens

id	int(11) AI		PK
createdAt	datetime		
description	varchar(191)	NULL	
expiresAt	datetime(3)		
jwtAccessToken	varchar(191)		
name	varchar(191)	NULL	
scopes	varchar(191)	NULL	
updatedAt	datetime(3)		
userId	int(11)		FK

approved-locations

id	int(11) AI	PK
createdAt	datetime	
subnet	varchar(191)	
userId	int(11)	FK

users

id	int(11) AI		PK
checkLocationOnLogin	tinyint(1)		
countryCode	varchar(191)		
createdAt	datetime		
gender	enum('male','female','nonbinary','unknown')		
name	varchar(191)		
nickname	varchar(191)		
notificationEmails	enum('account','updates','promotions')		
password	varchar(191)	NULL	
prefersColorScheme	enum('no_preference','light','dark')		
prefersLanguage	varchar(191)		
prefersReducedMotion	enum('no_preference','reduce')		
primaryEmail	int(11)	NULL	
profilePicture	varchar(191)		
role	enum('sudo','user')		
timezone	varchar(191)		
twoFactorEnabled	tinyint(1)		
twoFactorSecret	varchar(191)	NULL	
updatedAt	datetime(3)		
username	varchar(191)		AK

backup-codes

id	int(11) AI	PK
code	varchar(191)	
createdAt	datetime	
isUsed	tinyint(1)	
updatedAt	datetime(3)	
userId	int(11)	FK

sessions

id	int(11) AI		PK
city	varchar(191)	NULL	
countryCode	varchar(191)	NULL	
createdAt	datetime		
ipAddress	varchar(191)		
jwtToken	varchar(191)		
region	varchar(191)	NULL	
timezone	varchar(191)	NULL	
updatedAt	datetime(3)		
userAgent	varchar(191)		
userId	int(11)		FK

emails

id	int(11) AI	PK
createdAt	datetime	
email	varchar(191)	
isVerified	tinyint(1)	
updatedAt	datetime(3)	
userId	int(11)	FK

identities

id	int(11) AI	PK
createdAt	datetime	
identityId	varchar(191)	
loginName	varchar(191)	
type	varchar(191)	
updatedAt	datetime(3)	
userId	int(11)	FK

Figure 38: Database Entity Relationship Diagram

References

[1] "Refresh Tokens: When to Use Them and How They Interact with JWTs." [Online]. Available: https://auth0.com/blog/refresh-tokens-what-are-they-and-when-to-use-them/

[2] A. Vlasova, "7 Steps for effective software product development." [Online]. Available: https://relevant.software/blog/7-steps-for-effective-software-product-development-2018/

[3] G. Akrani, "Stages Process Steps of New Product Development." [Online]. Available: https://kalyan-city.blogspot.com/2012/02/stages-process-steps-of-new-product.html

[4] "Email statistics report, 2015-2019," The Radicati Group, Inc., Palo Alto, California, United States, Tech. Rep., Nov. 2019. [Online]. Available: https://www.radicati.com/wp/wp-content/uploads/2015/02/Email-Statistics-Report-2015-2019-Executive-Summary.pdf

[5] S. Blaszkiewicz, "Research Report: Online booking options can get you more clients," Mar. 2018. [Online]. Available: https://lab.getapp.com/research-online-booking-importance-of-appointment-scheduling/

[6] M. Dennis, "4 Ways AI Is Remaking Sales," Sep. 2017. [Online]. Available: https://x.ai/4-ways-ai-is-remaking-sales/

[7] ——, "How we calculate our AI's ROI," Apr. 2018. [Online]. Available: https://x.ai/how-we-calculate-our-ais-roi/

[8] J. Cranshaw, E. Elwany, T. Newman, R. Kocielnik, B. Yu, S. Soni, J. Teevan, and A. Monroy-Hernández, "Calendar.help: Designing a Workflow-Based Scheduling Agent with Humans in the Loop," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. Denver Colorado USA: ACM, May 2017, pp. 2382–2393. [Online]. Available: https://dl.acm.org/doi/10.1145/3025453.3025780

[9] J. Wajcman, "The Digital Architecture of Time Management," *Science, Technology, & Human Values*, vol. 44, no. 2, pp. 315–337, Mar. 2019. [Online]. Available: http://journals.sagepub.com/doi/10.1177/0162243918795041

[10] H. c. Editors, "Printing Press." [Online]. Available: https://www.history.com/topics/inventions/printing-press

[11] "Top 10 Free and Premium Calendaring Software in 2020 - Reviews, Features, Pricing, Comparison," Oct. 2019. [Online]. Available: https://www.predictiveanalyticstoday.com/top-calendaring-software/

[12] M. Masli, W. Geyer, C. Dugan, and B. Brownholtz, "The design and usage of tentative events for time-based social coordination in the enterprise," in *Proceedings of the 20th international conference on World wide web - WWW '11*. Hyderabad, India: ACM Press, 2011, p. 765. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1963405.1963512

[13] "Home Products & Services | UT wide products & services overview." [Online]. Available: https://www.utwente.nl/en/products-services/product/l829834/google-apps-ut-for-students

[14] S. F. Ehrlich, "Strategies for encouraging successful adoption of office communication systems," *ACM Transactions on Information Systems (TOIS)*, vol. 5, no. 4, pp. 340–357, Oct. 1987. [Online]. Available: http://dl.acm.org/doi/10.1145/42196.42198

[15] T. Erickson, C. M. Danis, W. A. Kellogg, and M. E. Helander, "Assistance: the work practices of human administrative assistants and their implications for it and organizations," in *Proceedings of the ACM 2008 conference on Computer supported cooperative work - CSCW '08*. San Diego, CA, USA: ACM Press, 2008, p. 609. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1460563.1460658

[16] N. Jennings and A. Jackson, "Agent-based meeting scheduling: A design and implementation," *Electronics Letters*, vol. 31, no. 5, pp. 350–352, Mar. 1995. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/el_19950245

[17] P. M. Berry, M. Gervasio, B. Peintner, and N. Yorke-Smith, "PTIME: Personalized assistance for calendaring," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 4, pp. 1–22, Jul. 2011. [Online]. Available: https://dl.acm.org/doi/10.1145/1989734.1989744

[18] J. F. Kelley and A. Chapanis, "How professional persons keep their calendars: Implications for computerization," *Journal of Occupational Psychology*, vol. 55, no. 4, pp. 241–256, Dec. 1982. [Online]. Available: http://doi.wiley.com/10.1111/j.2044-8325.1982.tb00098.x

[19] C. M. Kincaid, P. B. Dupont, and A. R. Kaye, "Electronic calendars in the office: an assessment of user needs and current technology," *ACM Transactions on Information Systems (TOIS)*, vol. 3, no. 1, pp. 89–102, Jan. 1985. [Online]. Available: http://dl.acm.org/doi/10.1145/3864.3868

[20] "Google Calendar for Android: Find a time for my meeting," Apr. 2016. [Online]. Available: https://blog.google/products/calendar/google-calendar-for-android-find-time/

[21] "AI Chatbots Try to Schedule Meetings—Without Enraging Us," *Wired*. [Online]. Available: https://www.wired.com/story/xai-meeting-ai-chatbot/

[22] "X.ai's AI meeting scheduler now costs $8 per month," Oct. 2018. [Online]. Available: https://venturebeat.com/2018/10/10/x-ai-introduces-calendar-view-and-new-plans-starting-at-8-per-month/

[23] "Clara Labs nabs $7M Series A as it positions its AI assistant to meet the needs of enterprise teams." [Online]. Available: https://social.techcrunch.com/2017/07/19/claraseriesa/

[24] D. Flanagan, *JavaScript: The Definitive Guide, 7th Edition.* O'Reilly Media, Inc., May 2020.

[25] "The State of JavaScript 2019: JavaScript Flavors." [Online]. Available: https://2019.stateofjs.com/javascript-flavors/

[26] G. Bierman, M. Abadi, and M. Torgersen, "Understanding TypeScript," in *ECOOP 2014 – Object-Oriented Programming*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, A. Kobsa, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, D. Terzopoulos, D. Tygar, G. Weikum, and R. Jones, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, vol. 8586, pp. 257–281. [Online]. Available: http://link.springer.com/10.1007/978-3-662-44202-9_11

[27] E. K. Kristensen and A. Møller, "Inference and Evolution of TypeScript Declaration Files," in *Fundamental Approaches to Software Engineering*, M. Huisman and J. Rubin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, vol. 10202, pp. 99–115. [Online]. Available: http://link.springer.com/10.1007/978-3-662-54494-5_6

[28] F. Kaimer and P. Brune, "Return of the JS: Towards a Node.js-Based Software Architecture for Combined CMS/CRM Applications," *Procedia Computer Science*, vol. 141, pp. 454–459, 2018. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1877050918317927

[29] S. Delcev and D. Draskovic, "Modern JavaScript frameworks: A Survey Study," in *2018 Zooming Innovation in Consumer Technologies Conference (ZINC).* Novi Sad: IEEE, May 2018, pp. 106–109. [Online]. Available: https://ieeexplore.ieee.org/document/8448444/

[30] "Svelte: Is This Next-Generation JavaScript Framework Production-Ready?" Aug. 2019. [Online]. Available: https://www.credera.com/blog/technology-solutions/svelte-is-this-next-generation-javascript-framework-production-ready/

[31] M. Li, J. Hu, and X. Lin, "The Development of Web Application Front-End of Intelligent Clinic Based on Vue.js," in *Proceedings of 2019 Chinese Intelligent Automation Conference*, Z. Deng, Ed. Singapore: Springer Singapore, 2020, vol. 586, pp. 683–690. [Online]. Available: http://link.springer.com/10.1007/978-981-32-9050-1_77

[32] "UT student among the top 50 young entrepreneurs." [Online]. Available: https://www.utoday.nl/news/64984/ut-student-in-the-top-50-young-entrepreneurs

[33] A. Chowdhary, "Staart." [Online]. Available: https://staart.js.org/

[34] "Stakeholder Analysis: Winning Support for Your Projects." [Online]. Available: http://www.mindtools.com/pages/article/newPPM_07.htm

[35] "Random User Generator | Home." [Online]. Available: https://randomuser.me/

[36] "This Person Does Not Exist." [Online]. Available: https://thispersondoesnotexist.com/

[37] "23-Year-Old's Design Collaboration Tool Figma Launches With $14M To Fight Adobe." [Online]. Available: https://social.techcrunch.com/2015/12/03/figma-vs-goliath/

[38] N. Ducheneaut and V. Bellotti, "E-mail as habitat: an exploration of embedded personal information management," *interactions*, vol. 8, no. 5, pp. 30–38, Sep. 2001. [Online]. Available: http://portal.acm.org/citation.cfm?doid=382899.383305

[39] A. de Barcelos Silva, M. M. Gomes, C. A. da Costa, R. da Rosa Righi, J. L. V. Barbosa, G. Pessin, G. De Doncker, and G. Federizzi, "Intelligent personal assistants: A systematic literature review," *Expert Systems with Applications*, vol. 147, p. 113193, Jun. 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0957417420300191

[40] J. Kiseleva, K. Williams, A. Hassan Awadallah, A. C. Crook, I. Zitouni, and T. Anastasakos, "Predicting User Satisfaction with Intelligent Assistants," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval - SIGIR '16*. Pisa, Italy: ACM Press, 2016, pp. 45–54. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2911451.2911521

[41] J. F. Kelley, "An iterative design methodology for user-friendly natural language office information applications," *ACM Transactions on Information Systems (TOIS)*, vol. 2, no. 1, pp. 26–41, Jan. 1984. [Online]. Available: http://dl.acm.org/doi/10.1145/357417.357420

[42] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, Jan. 2003. [Online]. Available: http://ieeexplore.ieee.org/document/1167344/

[43] C. A. Gomez-Uribe and N. Hunt, "The Netflix Recommender System: Algorithms, Business Value, and Innovation," *ACM Transactions on Management Information Systems*, vol. 6, no. 4, pp. 1–19, Jan. 2016. [Online]. Available: https://dl.acm.org/doi/10.1145/2843948

[44] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, Jun. 2005. [Online]. Available: http://ieeexplore.ieee.org/document/1423975/

[45] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, “Incorporating contextual information in recommender systems using a multidimensional approach,” *ACM Transactions on Information Systems (TOIS)*, vol. 23, no. 1, pp. 103–145, Jan. 2005. [Online]. Available: http://dl.acm.org/doi/10.1145/1055709.1055714

[46] A. Zunino and M. Campo, “Chronos: A multi-agent system for distributed automatic meeting scheduling,” *Expert Systems with Applications*, vol. 36, no. 3, pp. 7011–7018, Apr. 2009. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0957417408006040

[47] “Internet Calendaring and Scheduling Core Object Specification (iCalendar),” RFC Editor, Tech. Rep. RFC5545, Sep. 2009. [Online]. Available: https://www.rfc-editor.org/info/rfc5545

[48] T. Montal and Z. Reich, “I, Robot. You, Journalist. Who is the Author?: Authorship, bylines and full disclosure in automated journalism,” *Digital Journalism*, vol. 5, no. 7, pp. 829–849, Aug. 2017. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/21670811.2016.1209083

[49] K. N. Dörr and K. Hollnbuchner, “Ethical Challenges of Algorithmic Journalism,” *Digital Journalism*, vol. 5, no. 4, pp. 404–419, Apr. 2017. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/21670811.2016.1167612

[50] D. Catalanotto, “95% of the people stick to the default option,” Feb. 2019. [Online]. Available: https://service-design.co/95-of-the-people-stick-to-the-default-option-9e16316a64e1

[51] J. Pridmore and A. Mols, “Personal choices and situated data: Privacy negotiations and the acceptance of household Intelligent Personal Assistants:,” *Big Data & Society*, Jan. 2020. [Online]. Available: https://journals.sagepub.com/doi/10.1177/2053951719891748

[52] V. Kepuska and G. Bohouta, “Next-generation of virtual personal assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home),” *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, 2018.

[53] E. Zeng, S. Mare, and F. Roesner, “End user security & privacy concerns with smart homes,” in *Proceedings of the Thirteenth USENIX Conference on Usable Privacy and Security*, ser. SOUPS ’17. Santa Clara, CA, USA: USENIX Association, Jul. 2017, pp. 65–80.

[54] "This family's Echo sent a private conversation to a random contact." [Online]. Available: https://social.techcrunch.com/2018/05/24/family-claims-their-echo-sent-a-private-conversation-to-a-random-contact/

[55] "Why the Assistant Role is Female Dominated," Mar. 2018. [Online]. Available: https://teamels.com/why-the-assistant-role-is-female-dominated/

[56] PCMag, "The Real Reason Voice Assistants Are Female (and Why it Matters)," Jan. 2018. [Online]. Available: https://medium.com/pcmag-access/the-real-reason-voice-assistants-are-female-and-why-it-matters-e99c67b93bde

[57] "I'd blush if I could: closing gender divides in digital skills through education." [Online]. Available: https://unesdoc.unesco.org/ark:/48223/pf0000367416.page=85

[58] "ES modules: A cartoon deep-dive – Mozilla Hacks - the Web developer blog." [Online]. Available: https://hacks.mozilla.org/2018/03/es-modules-a-cartoon-deep-dive

[59] "The Twelve-Factor App." [Online]. Available: https://12factor.net/config

[60] "Lecture 5: Bayes Classifier and Naive Bayes." [Online]. Available: https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote05.html

[61] "8h30 / 8u.30 / 8.30 u. / 8.30 uur." [Online]. Available: https://taaladvies.net/taal/advies/vraag/1278/

[62] "Everything You Need To Know About Transactional Email But Didn't Know To Ask," Jun. 2018. [Online]. Available: https://www.smashingmagazine.com/2018/06/guide-transactional-email/

[63] "Daring Fireball: Markdown." [Online]. Available: https://daringfireball.net/projects/markdown/

[64] "@staart/mustache-markdown." [Online]. Available: https://www.npmjs.com/package/@staart/mustache-markdown

[65] "Nodemailer :: Nodemailer." [Online]. Available: https://nodemailer.com/about/

[66] N. Provos and D. Mazières, "A future-adaptive password scheme," in *Proceedings of the annual conference on USENIX Annual Technical Conference*, ser. ATEC '99. Monterey, California: USENIX Association, Jun. 1999, p. 32.

[67] "How To Safely Store A Password," Jan. 2010. [Online]. Available: https://codahale.com//how-to-safely-store-a-password/

[68] "Rate Limiting > Cisco Router Firewall Security: DoS Protection | Cisco Press." [Online]. Available: https://www.ciscopress.com/articles/article.asp?p=345618&seqNum=5

[69] "MariaDB/server," Jun. 2020, original-date: 2014-05-15T10:58:50Z. [Online]. Available: https://github.com/MariaDB/server

[70] "prisma/prisma," Jun. 2020, original-date: 2019-06-20T13:33:47Z. [Online]. Available: https://github.com/prisma/prisma

[71] A. E. Alami, M. Bahaj, and Y. Khourdifi, "Supply of a key value database redis in-memory by data from a relational database," in *2018 19th IEEE Mediterranean Electrotechnical Conference (MELECON).* Marrakech: IEEE, May 2018, pp. 46–51. [Online]. Available: https://ieeexplore.ieee.org/document/8379066/

[72] "Elastic Cloud: Hosted Elasticsearch, Hosted Search | Elastic." [Online]. Available: https://www.elastic.co/cloud/

[73] "elastic/elasticsearch," Jun. 2020, original-date: 2010-02-08T13:20:56Z. [Online]. Available: https://github.com/elastic/elasticsearch

[74] "Open Distro for Elasticsearch." [Online]. Available: https://opendistro.github.io/for-elasticsearch/

[75] E. Wittern, "Web APIs - challenges, design points, and research opportunities: invited talk at the 2nd international workshop on API usage and evolution (WAPI '18)," in *Proceedings of the 2nd International Workshop on API Usage and Evolution - WAPI '18.* Gothenburg, Sweden: ACM Press, 2018, pp. 18–18. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3194793.3194801

[76] T. Gupta, M. Sisodia, S. Fazulbhoy, M. Raju, and S. Agrawal, "Improving Accessibility for Dyslexic Impairments using Augmented Reality," in *2019 International Conference on Computer Communication and Informatics (ICCCI).* Coimbatore, Tamil Nadu, India: IEEE, Jan. 2019, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/document/8822152/

[77] alexchopin, "Announcing Nuxt's $2M seed round." [Online]. Available: https://nuxtjs.org/blog/seed-round

[78] A. Tosun, M. Ahmed, B. Turhan, and N. Juristo, "On the effectiveness of unit tests in test-driven development," in *Proceedings of the 2018 International Conference on Software and System Process - ICSSP '18.* Gothenburg, Sweden: ACM Press, 2018, pp. 113–122. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3202710.3203153

[79] Y. Li, J. Wang, Y. Yang, and Q. Wang, "An extensive study of class-level and method-level test case selection for continuous integration," *Journal of Systems and Software*, vol. 167, p. 110614, Sep. 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0164121220300923

[80] "What is CI? What is its purpose?" Sep. 2019. [Online]. Available: https://www.devlabsalliance.com/ci-and-its-purpose/

[81] S. Shokeen and A. Singh, "Deploying an e-commerce website using Amazon Web Services," in *2019 International Conference on contemporary Computing and Informatics (IC3I)*. Singapore, Singapore: IEEE, Dec. 2019, pp. 94–100. [Online]. Available: https://ieeexplore.ieee.org/document/9055586/

[82] S. Narula, A. Jain, and Prachi, "Cloud Computing Security: Amazon Web Service," in *2015 Fifth International Conference on Advanced Computing & Communication Technologies*. Haryana, India: IEEE, Feb. 2015, pp. 501–505. [Online]. Available: http://ieeexplore.ieee.org/document/7079135/

[83] "Oswald Labs Accelerator." [Online]. Available: https://oswaldlabs.com/accelerator/

[84] "Keep Your Website Always Online." [Online]. Available: https://www.cloudflare.com/always-online/

[85] "What is Uptime monitoring? Uses, Advantages, SEO Implications," Aug. 2018. [Online]. Available: https://www.marketing91.com/what-is-uptime-monitoring-uses-advantages-seo-implications/

[86] "About." [Online]. Available: https://uptimerobot.com/about

[87] "Netdata, a monitoring startup with 50-year-old founder, announces $17M Series A." [Online]. Available: https://social.techcrunch.com/2019/09/25/netdata-a-monitoring-startup-with-50-year-old-founder-announces-17m-series-a/

[88] "netdata/netdata," Jun. 2020, original-date: 2013-06-17T18:39:10Z. [Online]. Available: https://github.com/netdata/netdata

[89] "Sentry | Error Monitoring Software & Crash Reporting Tools for Apps." [Online]. Available: https://sentry.io/features/

[90] "Sentry raises $40 million Series C led by Accel for its error-tracking software." [Online]. Available: https://social.techcrunch.com/2019/09/24/sentry-raises-40-million-series-c-led-by-accel-for-its-error-tracking-software/

[91] "Sentry vs Logging." [Online]. Available: https://sentry.io/vs/logging/

[92] "Dashiki: Simple Request Breakdowns." [Online]. Available: https://analytics.wikimedia.org/dashboards/browsers/#all-sites-by-browser

[93] Z. Gu, C. Jin, D. Chang, and L. Zhang, "Predicting webpage aesthetics with heatmap entropy," *Behaviour & Information Technology*, pp. 1–15, Jan. 2020. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/0144929X.2020.1717626

[94] "2019 SBA Fact Sheet – Netherlands," Nov. 2019. [Online]. Available: https://ec.europa.eu/docsroom/documents/38662/attachments/21/translations/en/renditions/native

[95] Wedia, "Salary, minimum wage and payslips in the Netherlands." [Online]. Available: https://www.iamexpat.nl/career/working-in-the-netherlands/salary-payslip-dutch-minimum-wage