Student Housing: Roomies

Applied Research Document
S3-ITS

Lex de Kort

November 25, 2022

Version History

Version	Date	Changes	State
1.0	October 7, 2022	Initial release	Incomplete
2.0	November 11, 2022	Modified research questions & answers	Incomplete
3.0	November 25, 2022	Modified research questions & answers	Incomplete

Distribution

Version	Date	Receivers
1.0	October 7, 2022	Frank Coenen & Marcel Boelaars
2.0	November 4, 2022	Frank Coenen & Marcel Boelaars
3.0	November 25, 2022	Frank Coenen & Marcel Boelaars

Contents

1 Problem definition

As more and more people sign up for the housing platform, the available offers will increase. While this does mean that people have plenty of choice, it also means that there might be offerings that are simply not relevant to most users. Having to check the website every day only to find no suitable offers might eventually tire students. That, in turn, could lead to missed opportunities and offerings that are relevant to their needs.
Similarly, landlords with many listings and / or tenants would also have to spend more time than necessary manually monitoring the website to keep track of closed listings, messages, and other matters.
All these potential issues could inevitably result in users leaving the platform behind for greener pastures. To ensure user retention in the long-term, it is imperative that a solution is found to tackle these issues. To achieve this, applied research will be utilized.

2 Research Questions

In order to find a proper solution, we will summarize the problem in the form of a question:

How can *Roomies* keep its users properly and adequately informed by email about new listings that suit their preferences?

To answer this question, we will have to split the question up in separate smaller subquestions. They are as follows:

- **SQ-1: What libraries exist and are most used that we can use to keep users informed?**
- **SQ-2: What would users want to be informed about?**

Each of these questions will be tackled separately using various different research methods. Which research method will be used for a subquestion will depend on several factors:

- What the question entails
- The information an answer for a question requires
- The most applicable way to attain these answers

After identifying the best methods and analyzing the results, we can come to a conclusion that satisfies a given subquestion.

3 Methods utilized per subquestion

To find the answer for a subquestion two research methods will be utilized to triangulate.

SQ-1: What systems exist and are most used that we can use to keep users informed?

- Market analysis: what libraries are available for sending emails?
- Multi-criteria decision making: which library meets the necessary requirements for the service?

SQ-2: What information would users want to see on the website?

- Interview: interview students what they'd like to be informed about and how often they'd like to be informed
- Requirements prioritization: use the interview results to determine the most important requirements and modify existing requirements if need be

SQ-3: How will *Roomies* ensure that users get properly informed about the correct preferences?

- Prototyping: to test the viability of the chosen system, a small proof of concept will be created to simulate sending information
- Usability testing: after the proof of concept is created, we will test it against mock users to determine if the solution meets the requirements

4 Research and results

Now that we have established the subquestions and the methods we will use to answer them, it is time to start the actual research. Each subquestion will be answered on its own.

4.1 SQ-1: What systems exist and are most used that we can use to keep users informed?

4.1.1 Perform analysis on available email libraries built for Java

In order to implement an email service a solution is needed that can accommodate this functionality. Fortunately, there are libraries available that can cover these aspects. These libraries will also have to work on Java and be compatible with the Spring Boot framework.
Fortunately, it turns out that the Java Spring Boot framework comes with a built-in email library (0,). This, in turn, means we can simply use that instead of trying to find a different solution that may or may not be compatible with Spring Boot.

4.1.2 Choose a library based on a set of chosen criteria

As before, since Spring Boot comes with its own implementation it will be more convenient to use Spring Boot's solution for sending emails.

4.2 SQ-2: What would users want to be informed about?

4.2.1 Interview students to determine their preferences

In order to determine what information users would like to see, I conducted several interviews with potential end users of the application. From those interviews it was clear that most correspondents were at least satisfied with how information was displayed / given on other websites. There were still some aspects they didn't like as much, such as vagueness of the selection process or whether monthly costs for gas, water, light, and internet are included in the rent.
In terms of user requirements this means there is a lot of information that at first glance might not seem important but would still be nice to have some insight in.
Interviewees have indicated that having photos of the housing / room / apartment available on the website was helpful in getting a proper first impression. It also helped them imagine how they'd furnish the room.
The interviewees have also indicated that being able to communicate with their landlord through the website itself would be a nice functionality to have.
One point of frustration was how landlords would often defer to tenants to determine whether someone would be accepted or not. Another point of frustration is that new offerings are quickly taken. While *Roomies* cannot solve these issues directly, *Roomies* can offer extra tools for landlords that might reduce these frustations, such as different reponse strategies (for example: first-come-first-serve, lottery, etc.).

4.2.2 Analyse interview results to determine most important user requirements

To ensure that *Roomies* meets the needs of the potential end users, I will re-evaluate the user requirements that have already been determined.
After analyzing the interviews, the following information will be available on a given listing's details page, in no particular order:

- Monthly rent and breakdown of rent
- Surface area
- Location (address, city, neighborhood, etc.)
- Inclusion of furnishing
- Included accommodations (laundry machines, dish washer, etc.)
- Amount of (potential) roommates
- Allowance of pets
- A description of the house / room itself
- Images of the house / room

In terms of correspondence, interviewees have indicated that the most important information they want to see in e-mails is the price and location of listings with links to the details page of said listings.
Another aspect that will have to be implemented is image uploading. This will allow landlords to upload photo's and perhaps even floor plans of the housing. This in turn will result in a better user experience for *Roomies*' end users.
In terms of offering information, it is hard to determine how much information *Roomies* should be displaying.

5 Conclusion & recommendations

In conclusion, since we use Java Spring Boot e-mails are the best medium to send out information to our end users. Additionally, the end users have also indicated they prefer being e-mailed. As for what the e-mails should contain, end users have indicated their most preferred information they want to see is:

- Address
- City
- Neighborhood
- Rent

As such, the email contents will focus solely on displaying this information.
A potential proof of concept, as shown in *Figure 1*, shows what components such a (standalone) system would consist of.

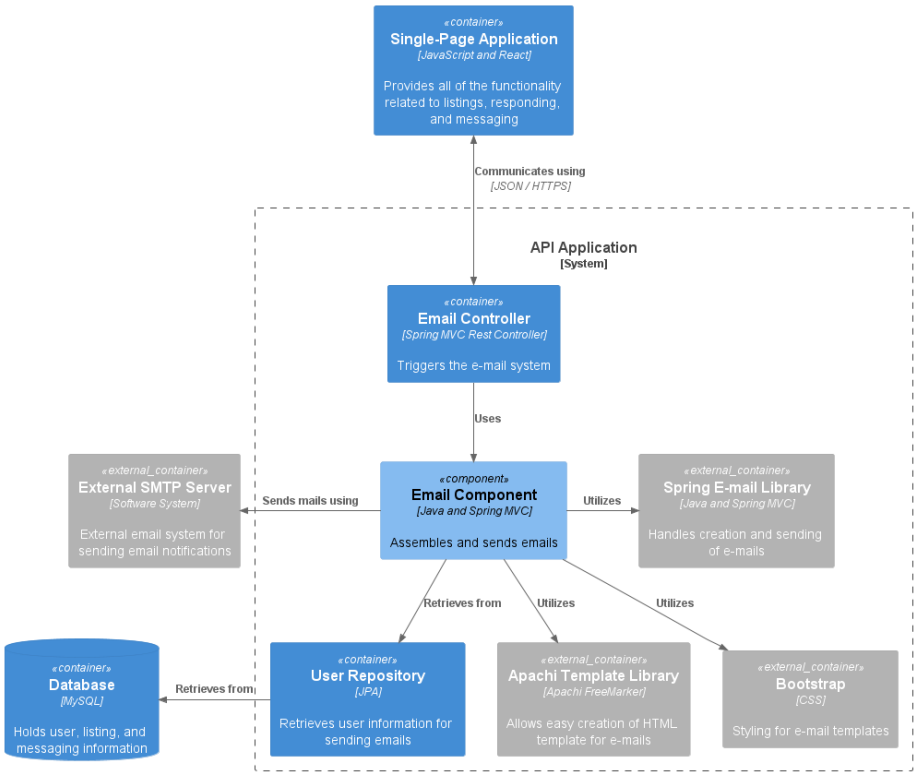

Figure 1: C3 Diagram

A prototype will also be developed based on the diagram that implements these features. It will in turn serve as proof of the concept being able to function

properly.

In conclusion, b

References

Baeldung. (2021, October 28). *Guide to Spring Email*. Baeldung. https://www.baeldung.com/spring-email