



CS 261 | SOFTWARE
DEVELOPMENT

W-IMs

An event management system

Table of Content

- Introduction
- Quick Overview
- How it Works!
- The magic behind it.
- Next Steps
- Overview of the semester

INTRODUCTION

Shouvik Ahmed Antu



Gregory Callahan



Douglas Li



Malie Heine



GOALS AND OBJECTIVES

Objective n° 1

Have a Final Project

Objective n° 2

Objective n° 3



GOALS AND OBJECTIVES

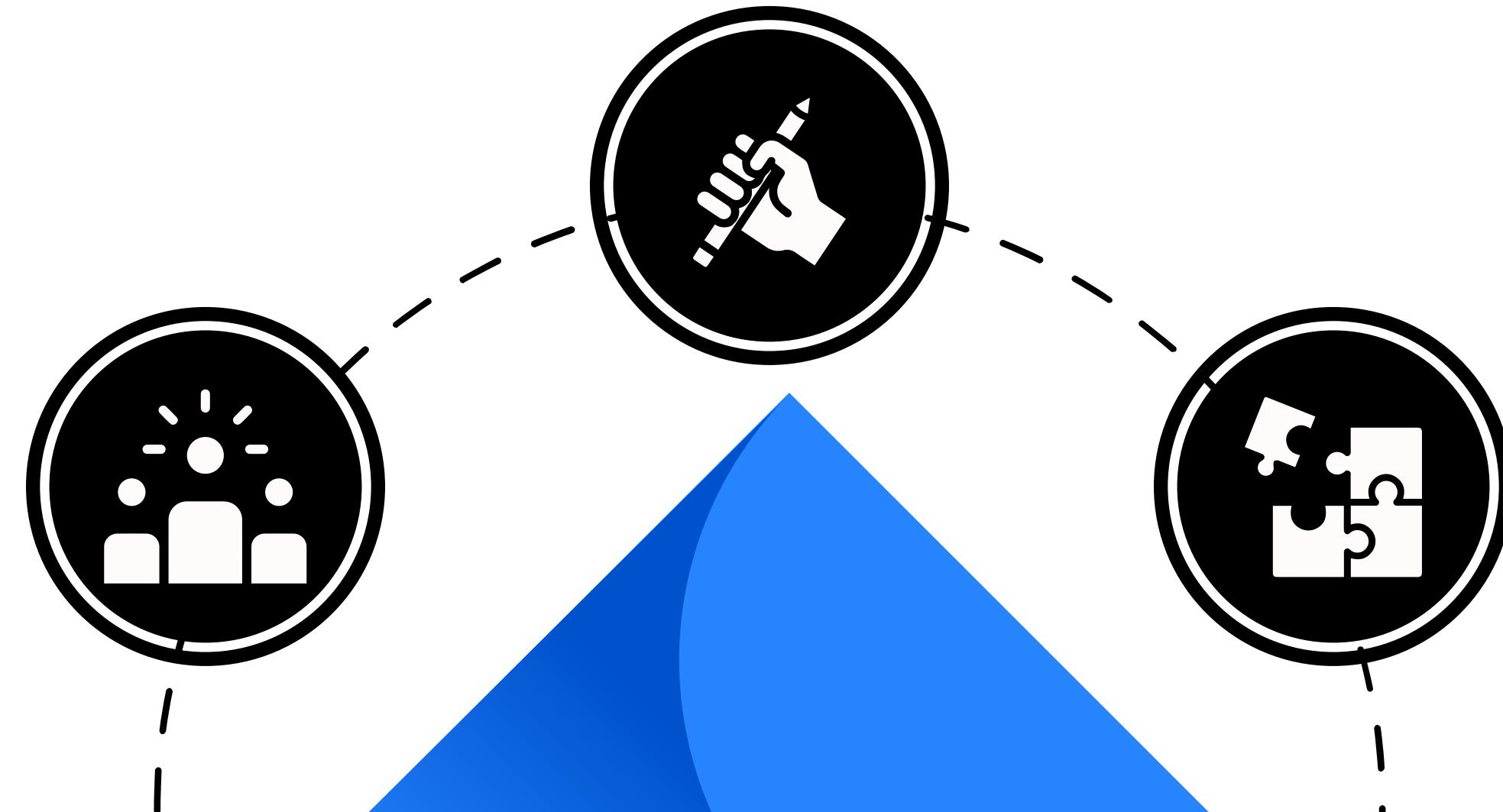
Objective n° 1

Have a Final Project

Objective n° 2

Have a Working Final Project

Objective n° 3



GOALS AND OBJECTIVES

Objective n° 1

Have a Final Project

Objective n° 2

Have a Working Final Project

Objective n° 3

Have a complete final
Project



GOALS AND OBJECTIVES

Objective n° 1

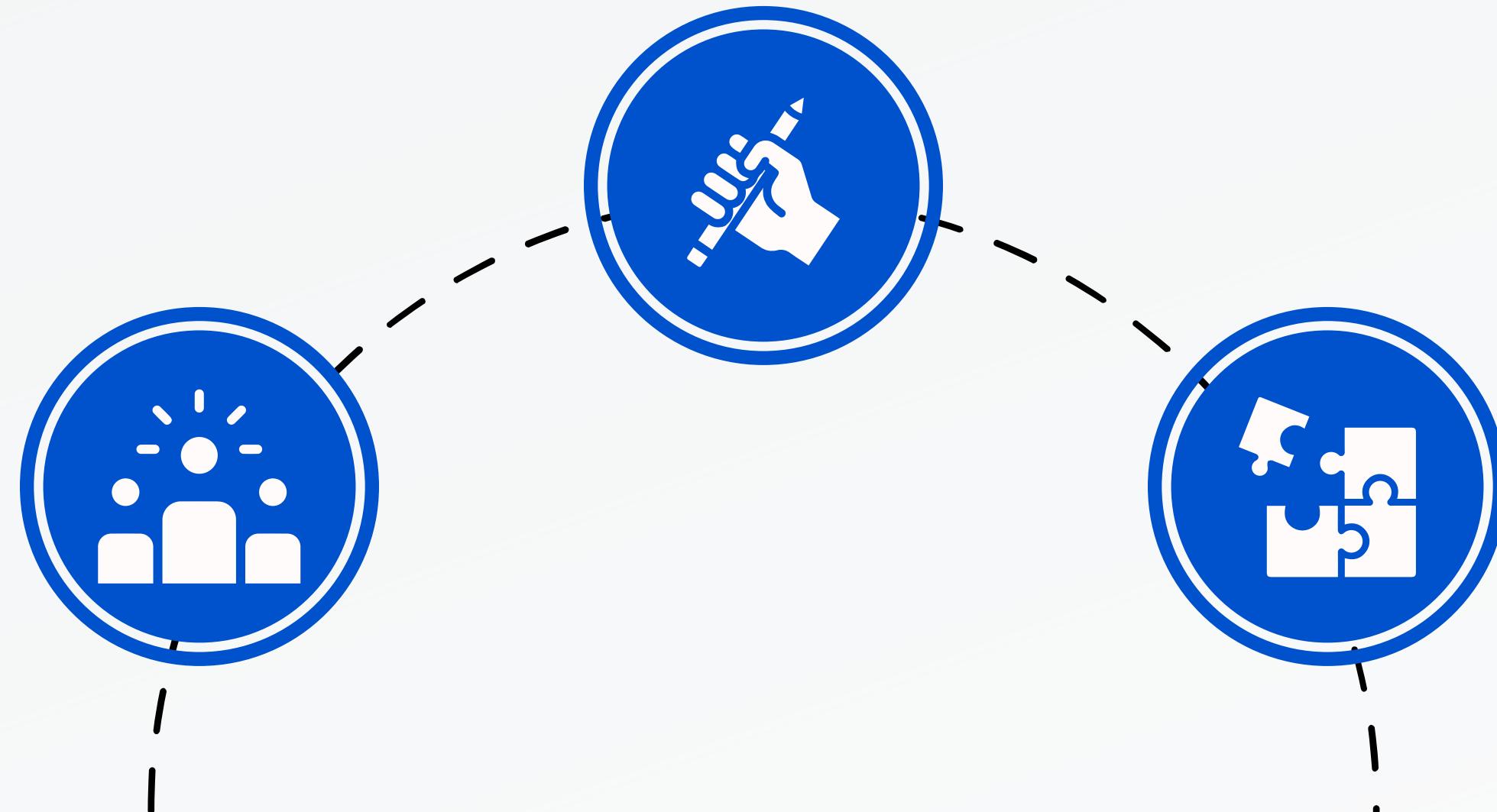
Something Useful. Can
be used in real-life.

Objective n° 2

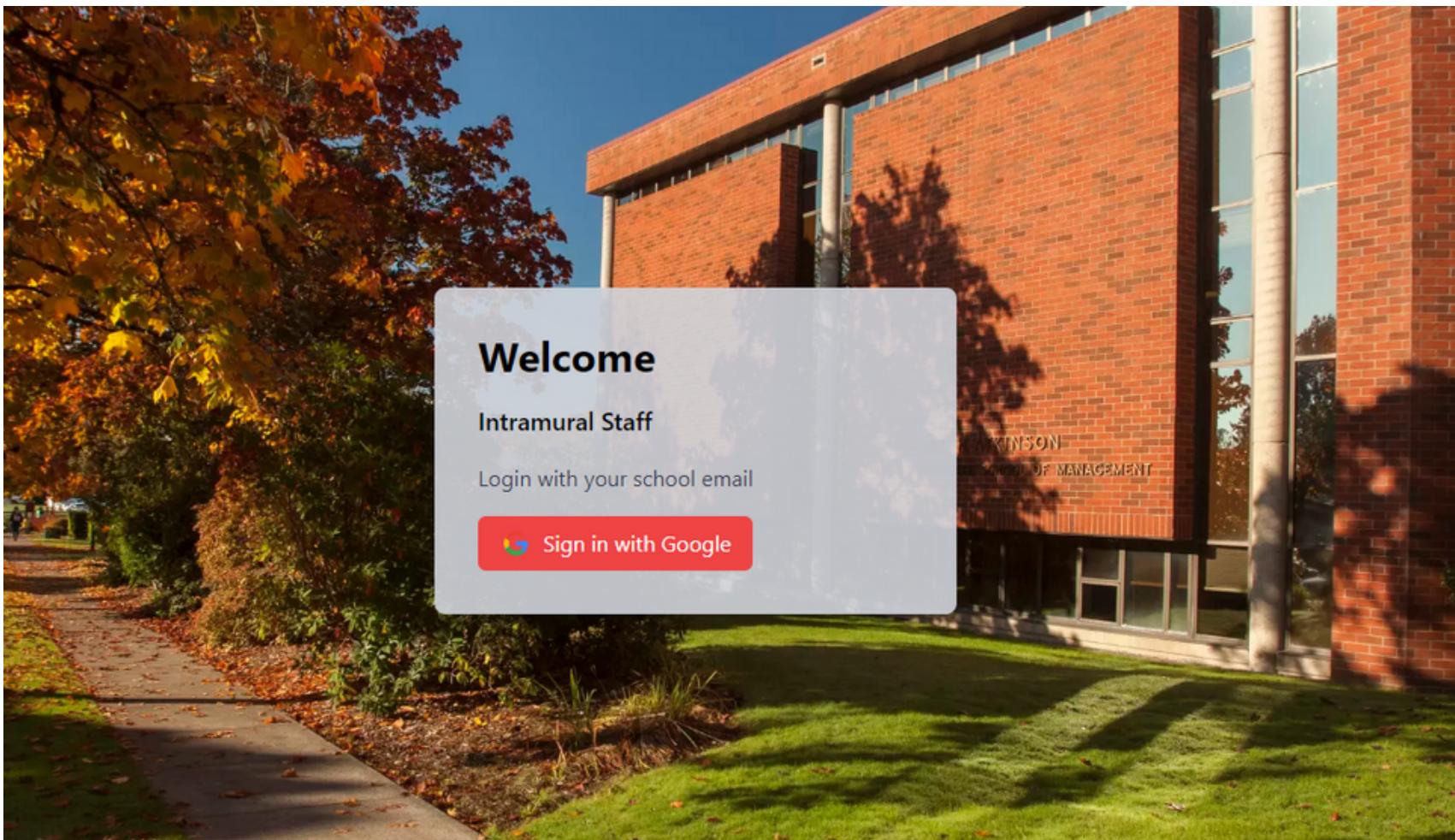
Learn How software development
actually works

Objective n° 3

Have a good & working
final Project



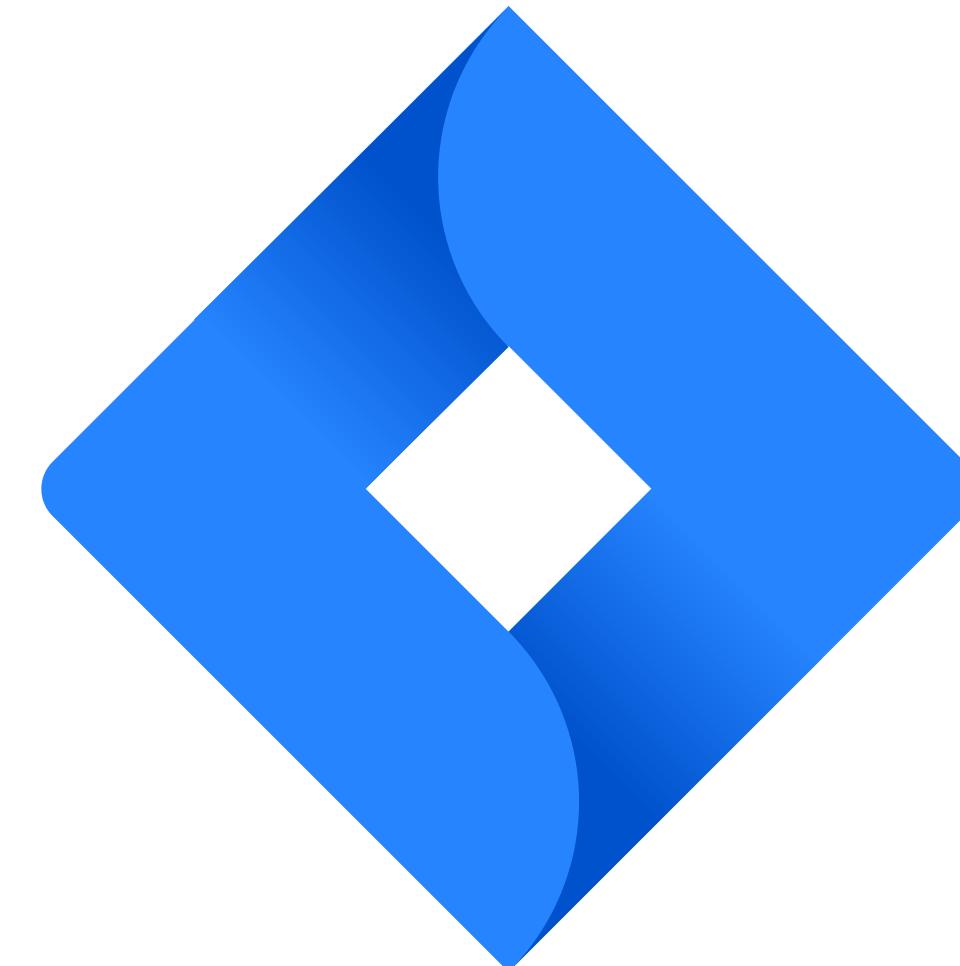
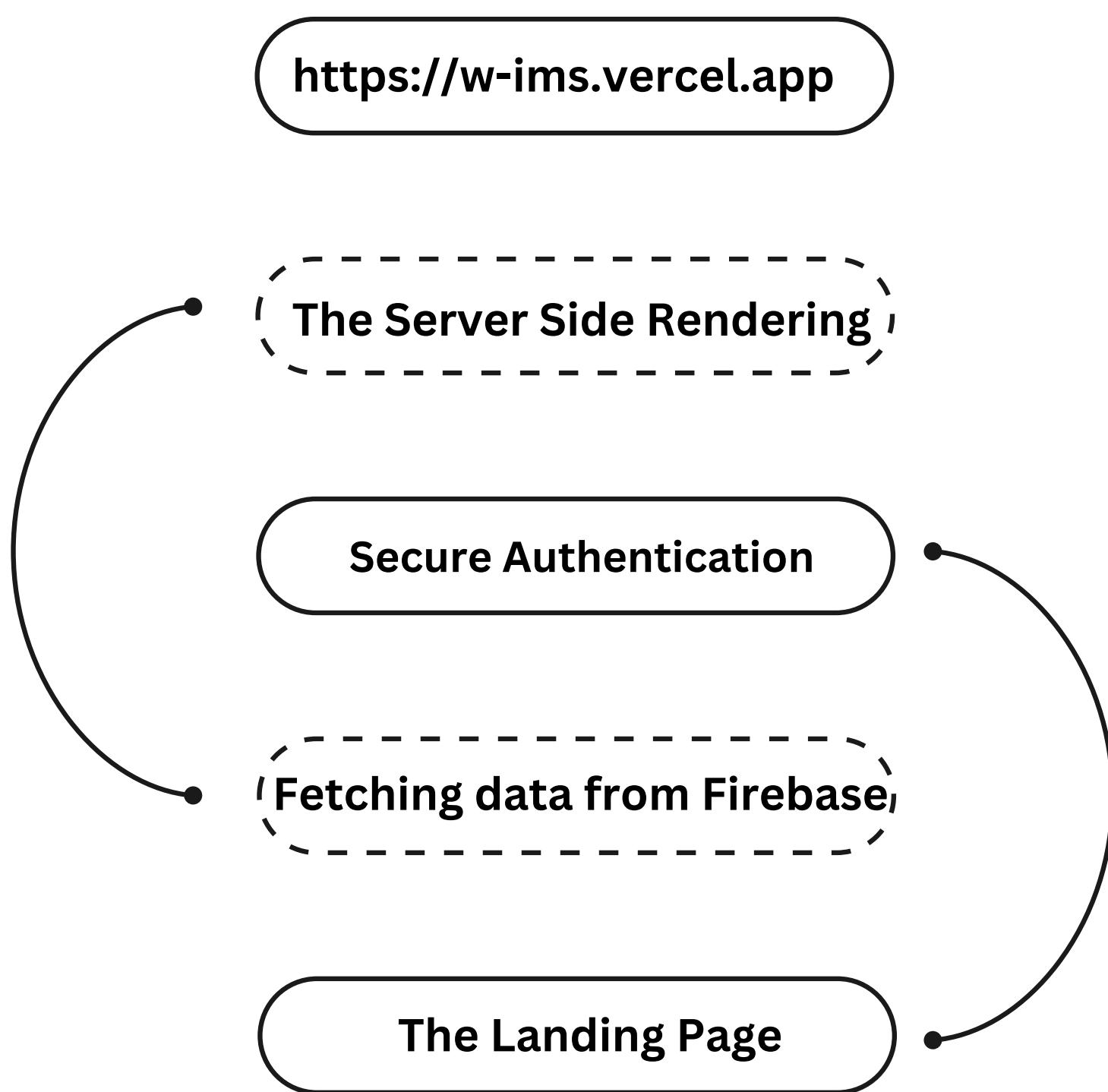
THE FULL TOUR



Existing System

The Better One

How it Works

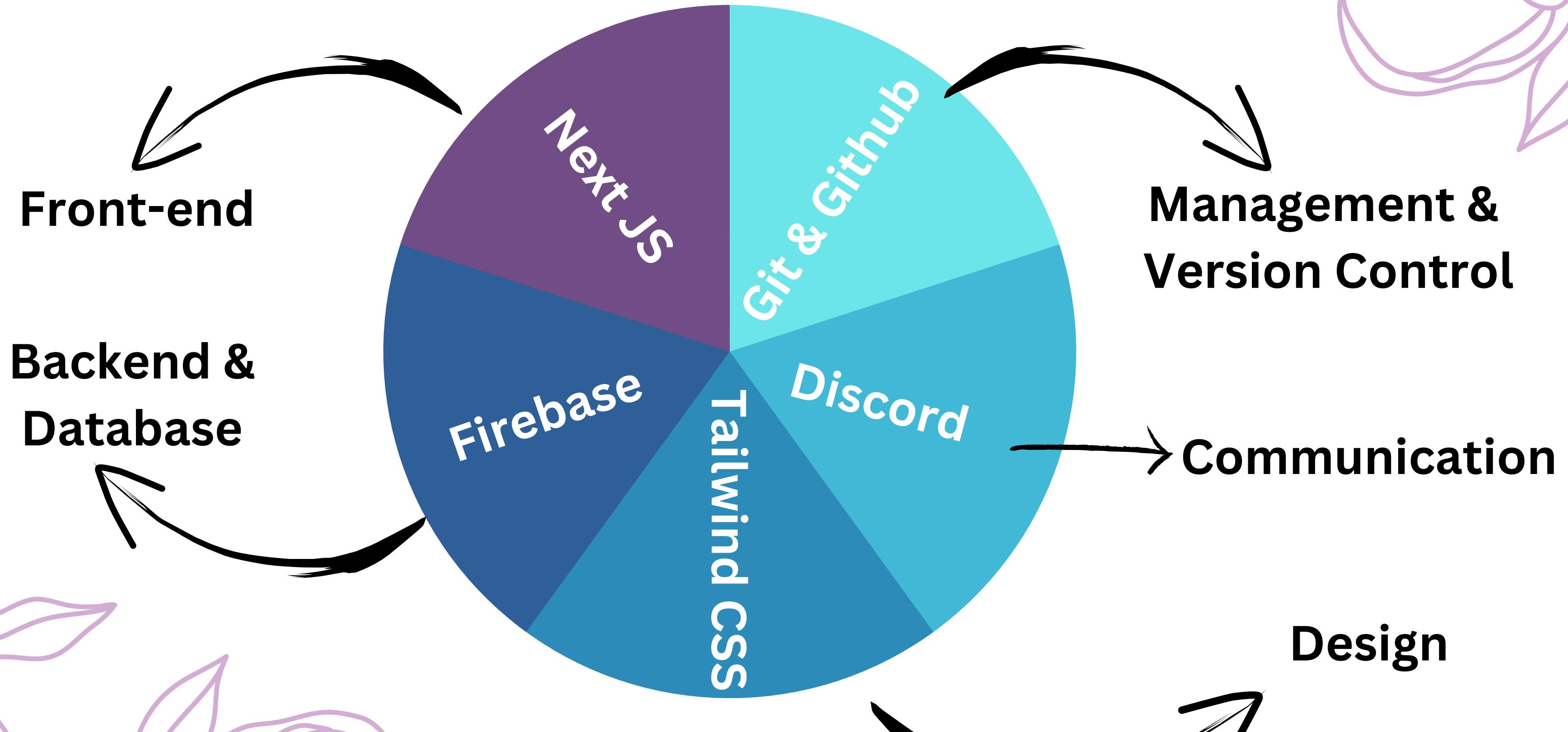


Built with a Plan

The Magic Behind it

Tech stacks & more







```
const handleLogin = (userData) => {
  setUser(userData);
  localStorage.setItem("user", JSON.stringify(userData)); // Store user in localstorage
};

const signIn = () => {
  signInWithPopup(auth, provider)
    .then((result) => {
      handleLogin(result.user); // call onLogin with user data
    })
    .catch((error) => alert(error.message));
};

return (
  <>
  {user ? (
    <div className="bg-container max-h-screen">
      <EventCreation user={user} />
    </div>
  ) : (
    <div className="bg-cover bg-center bg-container">
      <div className=" min-h-screen flex items-center justify-center">
        <div className="■ bg-gray-300 bg-opacity-90 p-8 rounded-lg shadow-lg max-w-sm w-full">
          <h1 className="text-3xl font-bold mb-4">Welcome</h1>
          <h2 className="text-lg font-medium mb-4">Intramural Staff</h2>
          <p className="■ text-gray-700 mb-4">
            Login with your school email
          </p>
        </div>
      </div>
    </div>
  )
}
```

- Enhanced performance
- Lazy loading(Dynamic)
- Security is of the highest priority.
- Static site generation

- 
- **Clean File Structure**
 - **Easy Navigation**
 - **Secure API end-points**

```
▽ components
  > assets
  JS edata.js
  JS EventCreation.js
  JS EventListing.js
  JS EventParticipants.js
  JS EventTable.js
  JS Header.js
▽ pages
  ▽ api
    ▽ auth
      JS [...nextauth].js
      JS hello.js
    ▽ registration
      JS [eventId].js
      JS _app.js
      JS _document.js
      JS edata.js
      JS events.js
      JS index.js
      JS success.js
    ▽ styles
      # globals.css
```

```
src
  components
    assets
    edata.js
    EventCreation.js
    EventListing.js
    EventParticipants.js
    EventTable.js
    Header.js
  pages
    api
      auth
        [...nextauth].js
        hello.js
      registration
        [eventId].js
        _app.js
        _document.js
        edata.js
        events.js
        index.js
        success.js
    styles
      # globals.css
      # Home.module.css
```

Component -based design

Api & Authentication

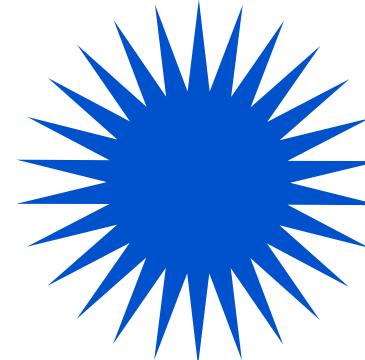
Easy Navigation

NextJs Authentication via Google sign in

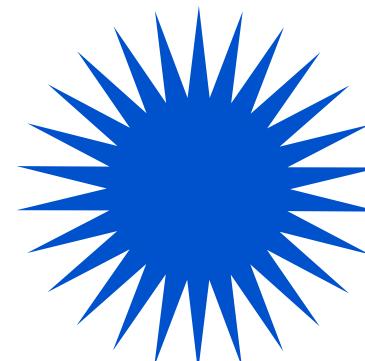
```
import NextAuth from "next-auth/next";
import GoogleProvider from "next-auth/providers/google";

export default NextAuth([
  providers: [
    GoogleProvider({
      clientId: process.env.GOOGLE_CLIENT_ID,
      clientSecret: process.env.GOOGLE_CLIENT_SECRET,
    }),
  ],
  callbacks: {
    async signIn({ user }) {
      if(user.email === "sahmed@willamette.edu" || user.email === "hcheng@willamette.edu" || user.email === "qnottage@willamette.edu" || user.email === "gcallahan@willamette.edu"){
        const isAllowedToSignIn = true
        if (isAllowedToSignIn) {
          return true
        } else {
          return alert("Unauthorized")
        }
      }
    },
    secret : process.env.NEXTAUTH_SECRET
  });
}
```

Firestore



Flexible Data Model



Scalable Database



Firebase





Cloud Firestore Data Model

Flexible data
Model

The diagram illustrates a hierarchical data structure in Cloud Firestore:

- rooms**: A collection containing:
 - roomA**: A document with field `name : "my chat room"`.
 - messages**: A collection containing:
 - message1**: A document with fields `from : "alex"` and `msg : "Hello World!"`.
 - message2**: A document (ellipsis).
- roomB**: A document (ellipsis).



Cloud Firestore Data Model

Flexible data Model

```
const EventParticipants = ({ ParentDocId }) => {
  const participantsRef = collection(db, "events", ParentDocId, "registrations")
  const [participants, setParticipants] = useState([])

  useEffect(() => {
    const participantscollection = onSnapshot(participantsRef, (snapshot) => {
      const newParticipants = snapshot.docs.map((doc) => ({
        id: doc.id,
        ...doc.data(),
      }));
      setParticipants(newParticipants);
    });
    return () => participantscollection
  }, []);
}
```



Cloud Firestore Data Model

Flexible data Model

Scalable

(events > AfmnsJgTYxtfB... > registrations)

events	AfmnsJgTYxtfBE3A5CQT	registrations
+ Add document	+ Start collection	+ Add document
6a1mbNpDgBIH9yZ3guL7	registrations	eCgRgpdmXnfiSdHRi09q
AfmnsJgTYxtfBE3A5CQT	>	pJXAXN63RHPvwZb8y7w4
S03ugwa0Hw8YR7s8D5jE		

+ Add field

eventDate: "2023-05-17"

eventDescription: "4 vs 4 volleyball tournament. Single Elimination."

eventLocation: "The Quad"

eventName: "Intramural Volleyball Tournament"



Cloud Firestore Data Model

Flexible data Model

Scalable

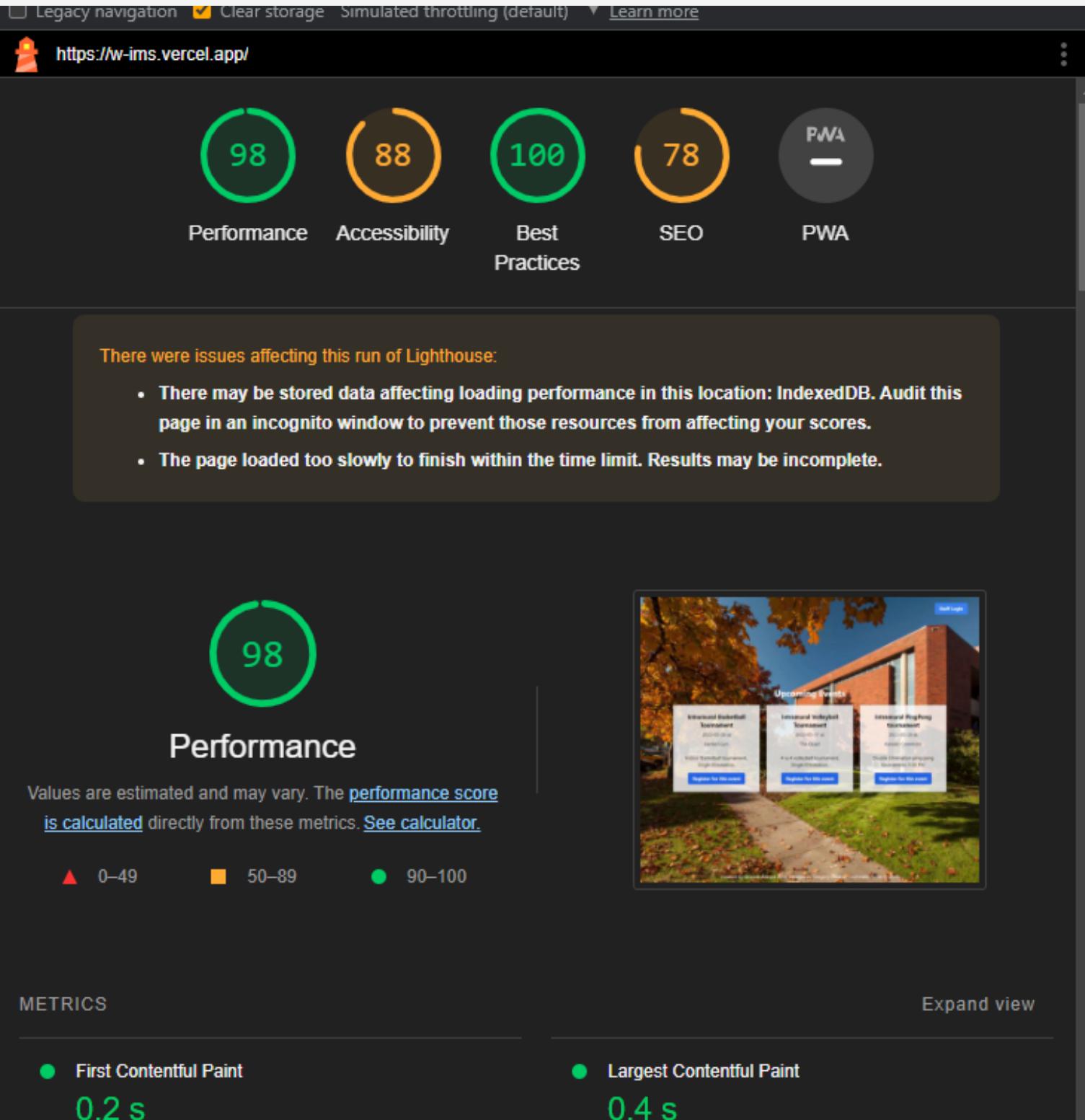
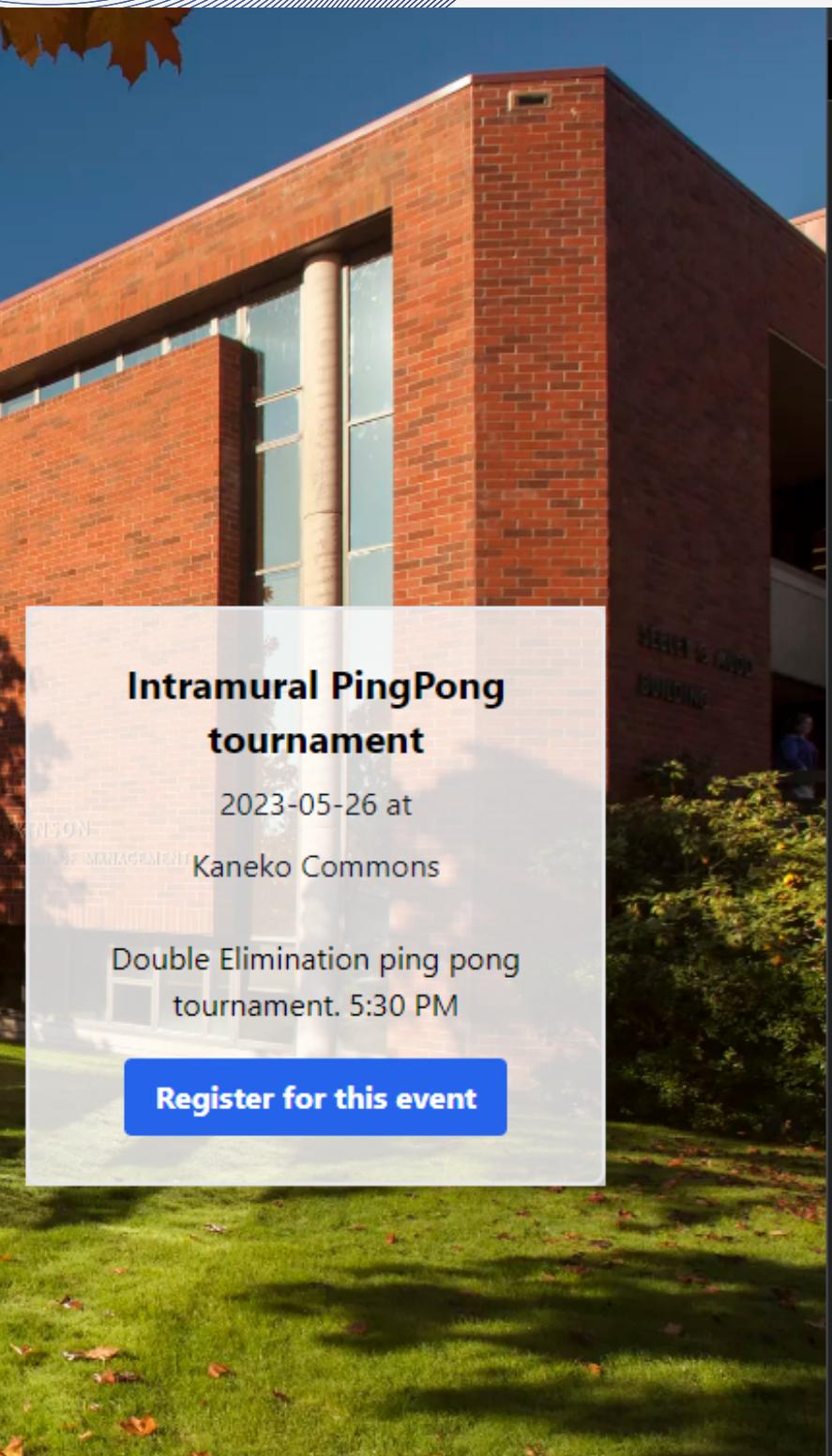
```
const EventCreation = ({ user }) => {
  const [eventName, set(eventName)] = useState("");
  const [eventDate, set(eventDate)] = useState("");
  const [eventLocation, set(eventLocation)] = useState("");
  const [eventDescription, set(eventDescription)] = useState("");

  const eventsRef = collection(db, "events");

  const handleSubmit = async (e) => {
    e.preventDefault();
    set(eventName(""));
    set(eventDate(""));
    set(eventLocation(""));
    set(eventDescription(""));
    alert("Event Successfully Created!");
  }

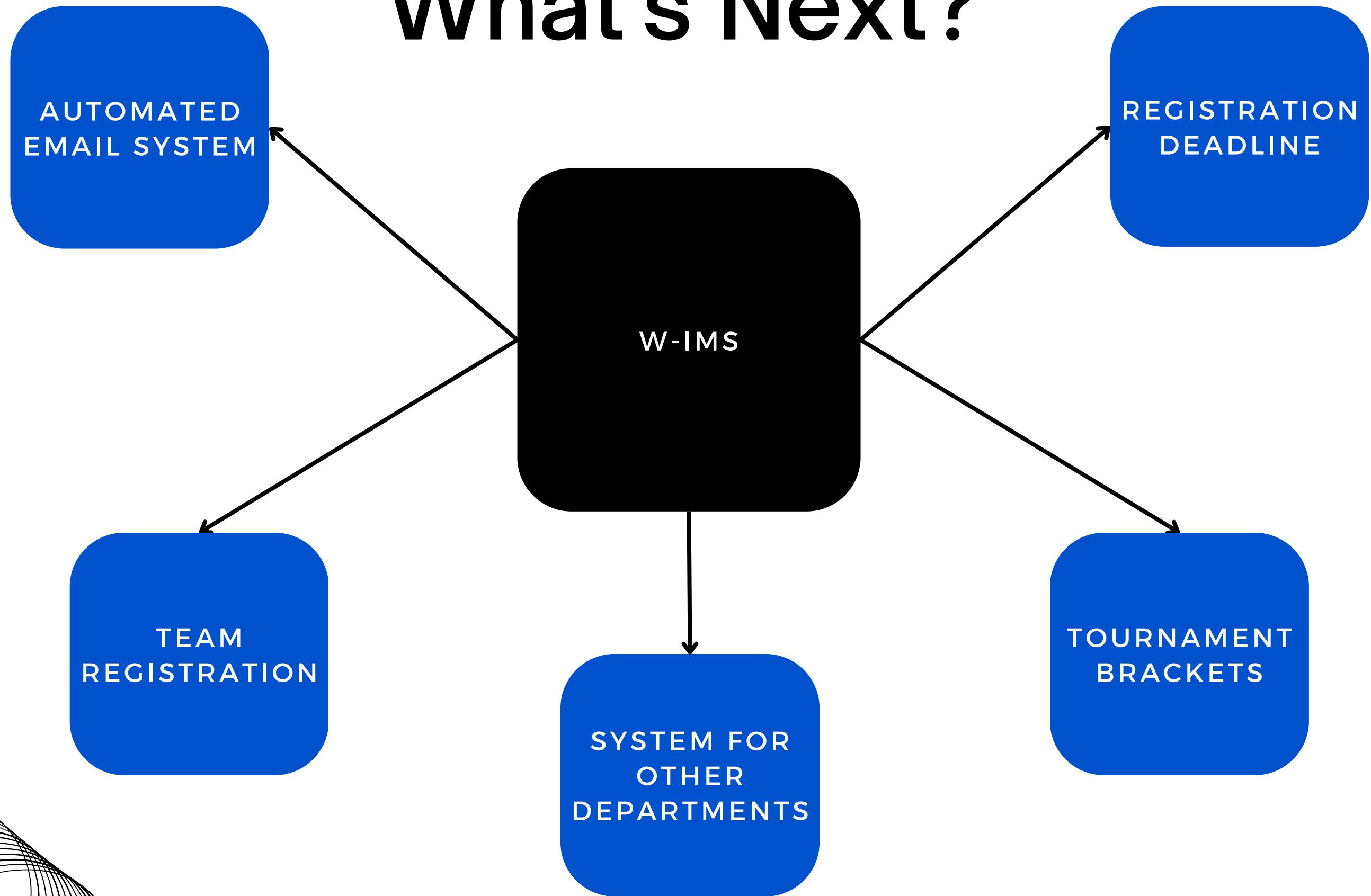
  const eventData = {
    eventName,
    eventDate,
    eventLocation,
    eventDescription,
  };

  try {
    await addDoc(eventsRef, eventData);
    // Handle success
  } catch (error) {
    // Handle error
  }
};
```



- Improved Search Engine Optimization
- Enhanced performance
- Lazy loading(Dynamic)
- Security is of the highest priority.
- Static site generation

What's Next?



Things that went well

- Explored options for project beyond class
- University likely to adopt system
- Small crunch time during SSRD
- Swapped tools when one wasn't working



Improvements going forward

- Meeting more frequently/regularly to update each other
- More frequent & clear communication
- Even distribution of Work



Overall Learning Outcome

- Software Development Methodologies - Agile Development
- Learned New technologies
- Do a lot of git commits
- Overall software development feel



The background features a large, abstract graphic composed of overlapping blue and white shapes. It includes a prominent white 'X' shape and several overlapping circles and triangles in varying shades of blue.

Thank You

Questions?