TaskFlow - Firebase and Vue.js Configuration Guide

🔧 Firebase Configuration

1. Firebase Project Setup

Create a Firebase Project

1. Go to Firebase Console
2. Click "Create a project" or "Add project"
3. Enter project name: `taskflow-management`
4. Choose your Google Analytics preferences
5. Wait for project creation to complete

Enable Required Services

```javascript
// Services to enable in Firebase Console:
- Authentication
- Firestore Database
- Hosting (optional for deployment)
```

2. Firebase Authentication Configuration

Enable Authentication Methods

1. Navigate to Authentication → Sign-in method
2. Enable the following providers:
   - Email/Password ✅
   - Google (optional) ✅

Firebase Auth Configuration

javascript

```javascript
// Firebase configuration object
const firebaseConfig = {
  apiKey: "your-api-key-here",
  authDomain: "taskflow-management.firebaseapp.com",
  projectId: "taskflow-management",
  storageBucket: "taskflow-management.appspot.com",
  messagingSenderId: "123456789012",
  appId: "1:123456789012:web:abcdef123456"
};

// Initialize Firebase
import { initializeApp } from 'firebase/app';
import { getAuth } from 'firebase/auth';
import { getFirestore } from 'firebase/firestore';

const app = initializeApp(firebaseConfig);
export const auth = getAuth(app);
export const db = getFirestore(app);
```

3. Firestore Database Structure

Collections Schema

javascript

```
// Users Collection
users: {
  [uid]: {
    email: string,
    displayName: string,
    role: 'admin' | 'team-member',
    createdAt: timestamp,
    lastLogin: timestamp
  }
}

// Tasks Collection
tasks: {
  [taskId]: {
    title: string,
    description: string,
    priority: 'low' | 'medium' | 'high',
    status: 'todo' | 'inprogress' | 'completed',
    assignee: string,
    assigneeId: string,
    dueDate: string,
    createdAt: timestamp,
    createdBy: string,
    createdById: string,
    updatedAt: timestamp
  }
}
```

Firestore Security Rules

```javascript
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    // Users can read/write their own user document
    match /users/{userId} {
      allow read, write: if request.auth != null && request.auth.uid == userId;
    }

    // All authenticated users can read all tasks
    match /tasks/{taskId} {
      allow read: if request.auth != null;
      allow create, update: if request.auth != null
        && request.auth.uid != null;
      allow delete: if request.auth != null
        && (resource.data.createdById == request.auth.uid
        || get(/databases/$(database)/documents/users/$(request.auth.uid)).data.role == 'admin');
    }
  }
}
```

🖥️ Vue.js Application Configuration

1. Project Structure

```
taskflow/
├── public/
│   ├── index.html
│   └── favicon.ico
├── src/
│   ├── components/
│   │   ├── Auth/
│   │   │   ├── LoginForm.vue
│   │   │   └── RegisterForm.vue
│   │   ├── Dashboard/
│   │   │   ├── DashboardStats.vue
│   │   │   ├── TaskForm.vue
│   │   │   ├── TaskList.vue
│   │   │   └── TaskFilters.vue
│   │   └── Layout/
│   │       ├── Header.vue
│   │       └── Sidebar.vue
│   ├── composables/
│   │   ├── useAuth.js
│   │   ├── useTasks.js
│   │   └── useFirestore.js
│   ├── services/
│   │   ├── firebase.js
│   │   └── api.js
│   ├── stores/
│   │   ├── auth.js
│   │   └── tasks.js
│   ├── router/
│   │   └── index.js
│   ├── App.vue
│   └── main.js
├── package.json
└── vite.config.js
```

2. Dependencies Installation

Core Dependencies

json

```
{
  "dependencies": {
    "vue": "^3.3.4",
    "vue-router": "^4.2.4",
    "pinia": "^2.1.6",
    "firebase": "^10.1.0",
    "@vueuse/core": "^10.2.1",
    "axios": "^1.4.0"
  },
  "devDependencies": {
    "@vitejs/plugin-vue": "^4.2.3",
    "vite": "^4.4.5",
    "tailwindcss": "^3.3.3",
    "autoprefixer": "^10.4.14",
    "postcss": "^8.4.24"
  }
}
```

Installation Commands

bash

```
# Create Vue.js project
npm create vue@latest taskflow-dashboard
cd taskflow-dashboard

# Install Firebase
npm install firebase

# Install additional dependencies
npm install @vueuse/core axios

# Install Tailwind CSS
npm install -D tailwindcss postcss autoprefixer
npx tailwindcss init -p
```

3. Vue.js Configuration Files

main.js

javascript

```
import { createApp } from 'vue'
import { createPinia } from 'pinia'
import router from './router'
import App from './App.vue'
import './firebase' // Initialize Firebase
import './style.css'

const app = createApp(App)
app.use(createPinia())
app.use(router)
app.mount('#app')
```

Vite Configuration (vite.config.js)

javascript

```
import { defineConfig } from 'vite'
import vue from '@vitejs/plugin-vue'
import path from 'path'

export default defineConfig({
  plugins: [vue()],
  resolve: {
    alias: {
      '@': path.resolve(__dirname, './src'),
    },
  },
  server: {
    port: 3000,
  },
  build: {
    outDir: 'dist',
    assetsDir: 'assets',
  }
})
```

4. Environment Variables

.env.local

```bash
# Firebase Configuration
VITE_FIREBASE_API_KEY=your-api-key-here
VITE_FIREBASE_AUTH_DOMAIN=taskflow-management.firebaseapp.com
VITE_FIREBASE_PROJECT_ID=taskflow-management
VITE_FIREBASE_STORAGE_BUCKET=taskflow-management.appspot.com
VITE_FIREBASE_MESSAGING_SENDER_ID=123456789012
VITE_FIREBASE_APP_ID=1:123456789012:web:abcdef123456

# Application Configuration
VITE_APP_TITLE=TaskFlow Dashboard
VITE_APP_VERSION=1.0.0
```

🚀 Development Workflow

1. Development Commands

```bash
# Start development server
npm run dev

# Build for production
npm run build

# Preview production build
npm run preview

# Run tests
npm run test

# Lint code
npm run lint
```

2. Firebase Deployment

Install Firebase CLI

```bash
npm install -g firebase-tools
firebase login
firebase init hosting
```

Firebase Hosting Configuration (firebase.json)

json

```
{
  "hosting": {
    "public": "dist",
    "ignore": [
      "firebase.json",
      "**/.*",
      "**/node_modules/**"
    ],
    "rewrites": [
      {
        "source": "**",
        "destination": "/index.html"
      }
    ]
  }
}
```

Deployment Commands

bash

```
# Build and deploy
npm run build
firebase deploy --only hosting

# Deploy with custom message
firebase deploy --only hosting -m "Version 1.0.0 deployment"
```

🔒 Security Configuration

1. Authentication Security

javascript

```javascript
// Auth state persistence
import { setPersistence, browserLocalPersistence } from 'firebase/auth';

setPersistence(auth, browserLocalPersistence)
  .then(() => {
    // Persistence set successfully
  })
  .catch((error) => {
    console.error('Auth persistence error:', error);
  });
```

2. API Security

javascript

```javascript
// Request interceptors for API calls
import axios from 'axios';
import { getAuth } from 'firebase/auth';

const api = axios.create({
  baseURL: 'https://api.taskflow.com',
  timeout: 10000,
});

// Add auth token to requests
api.interceptors.request.use(async (config) => {
  const auth = getAuth();
  if (auth.currentUser) {
    const token = await auth.currentUser.getIdToken();
    config.headers.Authorization = `Bearer ${token}`;
  }
  return config;
});
```

📊 Performance Optimization

1. Code Splitting

```javascript
// Lazy load components
const Dashboard = () => import('@/components/Dashboard/Dashboard.vue');
const TaskManagement = () => import('@/components/Tasks/TaskManagement.vue');

const routes = [
  {
    path: '/dashboard',
    component: Dashboard
  },
  {
    path: '/tasks',
    component: TaskManagement
  }
];
```

2. Firebase Performance Monitoring

```javascript
import { getPerformance } from 'firebase/performance';

// Initialize Performance Monitoring
const perf = getPerformance(app);
```

🧪 Testing Configuration

1. Unit Testing Setup

```bash
# Install testing dependencies
npm install -D @vue/test-utils vitest jsdom
```

2. Test Configuration (vitest.config.js)

```javascript
import { defineConfig } from 'vitest/config'
import vue from '@vitejs/plugin-vue'

export default defineConfig({
  plugins: [vue()],
  test: {
    environment: 'jsdom',
    globals: true,
  },
})
```

📝 Additional Configuration Notes

- **Real-time Updates**: Configure Firestore listeners for real-time task updates
- **Offline Support**: Implement service worker for offline functionality
- **PWA Features**: Configure manifest.json for Progressive Web App capabilities
- **Error Handling**: Set up global error handling and logging
- **Analytics**: Configure Google Analytics for user behavior tracking

🔧 Troubleshooting

Common Issues

1. **Firebase Configuration**: Ensure all environment variables are properly set
2. **CORS Issues**: Configure Firebase security rules correctly
3. **Build Errors**: Check for missing dependencies or configuration issues
4. **Authentication**: Verify Firebase project settings and API keys

Debug Mode

```javascript
// Enable Firebase debug mode in development
if (import.meta.env.DEV) {
  // Enable debug logging
  console.log('Firebase debug mode enabled');
}
```